

AD-A089 109

TRANSPORTATION SYSTEMS CENTER CAMBRIDGE MA F/6 4/2
VOICE RESPONSE SYSTEM STATISTICS PROGRAM, OPERATIONAL HANDBOOK. (U)
JUN 80 I ENGLANDER
FCA-FAA-80-9 FAA-80-80-76 NI

F/6 4/2

UNCLASSIFIED

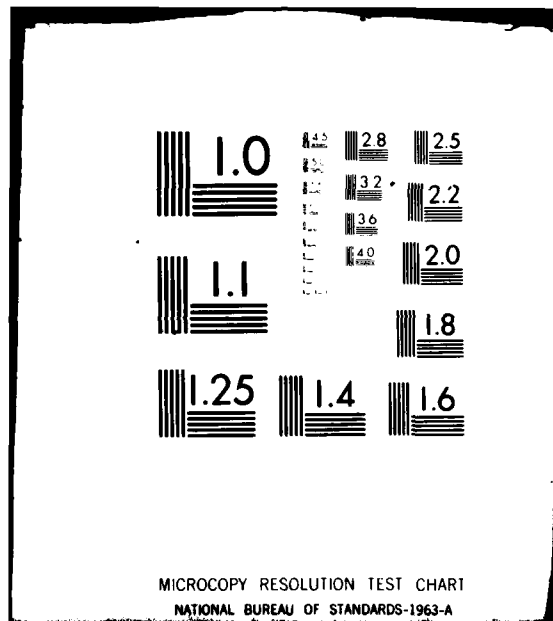
TSC-FAA-80-9

FAA-RD-80-76

NL

100

END
DATE
FILMED
10-80
DTIC



REPORT NO. FAA-RD-80-76

LEVEL *II*

13
P.S.

VOICE RESPONSE SYSTEM STATISTICS PROGRAM
Operational Handbook

Irwin Englander

U.S. DEPARTMENT OF TRANSPORTATION
RESEARCH AND SPECIAL PROGRAMS ADMINISTRATION
Transportation Systems Center
Cambridge MA 02142

AD A089109



JUNE 1980
FINAL REPORT

DTIC
ELECTRONIC
SEP 11 1980
E

DOCUMENT IS AVAILABLE TO THE PUBLIC
THROUGH THE NATIONAL TECHNICAL
INFORMATION SERVICE, SPRINGFIELD,
VIRGINIA 22161

Prepared for
U.S. DEPARTMENT OF TRANSPORTATION
FEDERAL AVIATION ADMINISTRATION
Systems Research and Development Service
Washington DC 20591

DC FILE COPY

80 9 8 038

NOTICE

This document is disseminated under the sponsorship of the Department of Transportation in the interest of information exchange. The United States Government assumes no liability for its contents or use thereof.

NOTICE

The United States Government does not endorse products or manufacturers. Trade or manufacturers' names appear herein solely because they are considered essential to the object of this report.

Technical Report Documentation Page

1. Report No. 18 FAA-RD-80-76 ✓	2. Government Accession No. AD-A089 109	3. Recipient's Catalog No. 12701
4. Title and Subtitle VOICE RESPONSE SYSTEM STATISTICS PROGRAM, Operational Handbook	5. Report Date 11 June 1980	6. Performing Organization Code DTS-533
7. Author(s) 10 Irwin/Englander	8. Performing Organization Report No. 14 DOT-TSC-FAA-80-91	9. Performing Organization Name and Address U.S. Department of Transportation Research and Special Programs Administration Transportation Systems Center ✓ Cambridge MA 02142
10. Sponsoring Agency Name and Address U.S. Department of Transportation Federal Aviation Administration Systems Research and Development Service Washington DC 20591	11. Work Unit No. (TRAIS) FA031/R0101	12. Type of Report and Period Covered 9 Final Report April 1978-December 1979
13. Supplementary Notes		14. Sponsoring Agency Code ARD-421
16. Abstract This report documents the Voice Response System (VRS) Statistics Program developed for the preflight weather briefing VRS. It describes the VRS statistical report format and contents, the software program structure, and the program operation.		
17. Key Words VRS, VRS Statistics		18. Distribution Statement DOCUMENT IS AVAILABLE TO THE PUBLIC THROUGH THE NATIONAL TECHNICAL INFORMATION SERVICE, SPRINGFIELD, VIRGINIA 22161.
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 76
		22. Price

PREFACE

The VRS statistics program was developed by the Transportation Systems Center under the guidance of the Federal Aviation Administration sponsors, C. Weigel and V. Costantino. The preflight weather briefing VRS software and VRDATA.DAT file were developed under contract by Input Output Computer Services (IOCS), Waltham MA.

Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DDC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification _____	
By _____	
Distribution/ _____	
Availability Codes	
Dist.	Avail and/or special
A	

PROCESSING PAGE BLANK-NOT FILMED

METRIC CONVERSION FACTORS

Approximate Conversions to Metric Measures			
Symbol	When You Know	Multiply by	To Find
m cm mm km	LENGTH		
	meters	0.30	yards
	centimeters	0.4	inches
	millimeters	2.5	inches
m ² cm ² km ²	AREA		
	square meters	1.2	square yards
	square centimeters	0.4	square inches
	square kilometers (10,000 m ²)	2.5	square miles
g kg tonne (1,000 kg)	MASS (weight)		
	grams	0.005	ounces
	kilograms	2.2	pounds
	tonnes	1.1	short tons
l ml m ³ km ³	VOLUME		
	milliliters	0.005	fluid ounces
	liters	1.1	pints
	cubic meters	35	cubic feet
°C	TEMPERATURE (Celsius)		
	Celsius temperature	9/5 (then add 32)	Fahrenheit temperature



TABLE OF CONTENTS

<u>Section</u>	<u>Page</u>
1. INTRODUCTION.....	1
2. VRS STATISTICAL REPORT DESCRIPTION.....	2
2.1 The Header.....	2
2.2 User Report.....	2
2.3 Summary Report.....	3
3. ANALYSIS OF THE VRS STATISTICAL FILE.....	4
4. PROGRAM STRUCTURE.....	7
5. STATISTICS PROGRAM OPERATION.....	33
APPENDIX A - SAMPLE VRDATA.DAT FILE (PARTIAL DUMP - 1st 3 BLOCKS).....	35
APPENDIX B - SAMPLE VRS STATISTICAL REPORT.....	40
APPENDIX C - PROGRAM LISTING.....	42

LIST OF ILLUSTRATIONS

<u>Figure</u>	<u>Page</u>
4-1. MAIN PROGRAM AND DIRECTLY CALLED PROGRAMS.....	9
4-2. PROGRAMS USED TO READ THE RAW DATA FILE.....	11
4-3. ALL ROUTINES WHICH DETERMINE WHEN A USER HAS LOGGED ON AN OFF.....	14
4-4. LOCATION-ORIENTED OPTIONS.....	20

LIST OF TABLES

<u>Table</u>	<u>Page</u>
3-1. FILE VRDATA.DAT RECORD FORMAT.....	4
3-2. US.KEY (HIGH-ORDER BYTE).....	5
3-3. US.FLG (LOW-ORDER BYTE).....	5
3-4. SPECIAL FUNCTION CODES (US.STA - LOW-ORDER BYTE).....	6
4-1. TEST TABLE.....	23
5-1. OPTION TABLE.....	33

1. INTRODUCTION

An automated preflight weather briefing voice response system (VRS) developed by the Transportation Systems Center¹ is currently undergoing continuous operational testing and evaluation by the aviation community in the Washington DC and Columbus OH areas. During each VRS operating day, raw statistical data is stored in a file named VRDATA.DAT. This file is the basis of the statistical report. A description of the VRS statistical program and the report is presented in the following sections.

2. VRS STATISTICAL REPORT DESCRIPTION

The VRS statistical report consists of 3 parts: the header, the user report, and the daily summary. Each will be described below.

2.1 THE HEADER

The header page is always printed regardless of the output option selected. It contains the following information:

Minimum Channel No. - a 2 digit number between 00 and 20
Maximum Channel No. - same as above.
Start Point of File:

Start Date - year, month, date (YY MM DD)
Block No. - starting data block
Byte Offset - starting data byte within the starting block
Time - minutes since midnight (GMT)

End Point of File:

Finish Date - year, month, day (YY MM DD)
Block No. - last data block
Byte No. - last data byte within the last data block
Time - minutes since midnight (GMT)

2.2 USER REPORT

The User Report has 1 line per user. There are 18 columns, divided into 5 groups. The first group has 1 column, GMT time. This is the time of the user channel disconnect.

Group 2 has 4 columns, each of which is a count of user requests. Column LOC is a count of location identifiers entered. Column SA is a count of Hourly Surface Observation (SA) reports requested. Column FT is a count of Terminal Forecast (FT) reports requested. Column GF is a count of winds alofts (GF) reports requested.

Group 3 has 6 columns and shows the allocation of user time. Column TIME shows the total time allocated to the user. Column OVER shows the time allocated to overhead. Column LOC shows the time allocated to entering location identifiers. This is from user entry of the first letter until the message is sent to the PDP 11-70 and accepted by the PDP 11-70. Column SA is the time from the SA request until all SA's have been spoken. Column FT is the time from a request for FT reports until all FT reports are spoken. Column GF is the time from the GF request until all the GF reports are spoken.

Group 4 has 6 columns, which contain the counts of how many times the user used the special functions.

Special Functions

<u>Title</u>	<u>Function</u>
Stop	Stop voice output
Go	Proceed with voice output
RPT	Repeat prompt or current report
DEL	Cancel last user entry
JMP	Jump to next report
BEG	Recycle to beginning of briefing.

The last column, which is labelled, LOC ID REQUESTED contains a list of location identifiers entered by the user during the last pass through the briefing.

2.3 SUMMARY REPORT

The fourth report type is a summary of the day's activity. The first column is the hour in GMT. The second column is the number of users during the hour. The third column is the average time used by each user. The fourth column is the maximum number of simultaneous users during the hour.

Below this is the total number of users. This is after deleting users who merely called in then hung up.

The average time is the total time by all users, divided by the number of users. Next is the maximum number of simultaneous users for the entire day.

The fractional percentages are found by summing the times for all users and dividing the time for each purpose by the total time users were on the system. OVHR is the percentage of time allocated to overhead (i.e., the system is neither accepting user inputs nor responding to user requests. It is derived from the column OVER in the User Report. Similarly LOCID is derived from the LOC values, and SA, FT, GF from their corresponding values in the User Report.

The last part of the daily summary report contains a list of all location identifiers requested and the request counts.

3. ANALYSIS OF THE VRS STATISTICAL FILE

During each VRS operating day, a statistics file* is maintained (see Appendix A). In block 0 of the file is the starting date and time in seconds and the finishing data and time in seconds. The first page of the statistical report contains these data as date and time in minutes for identification (see Appendix B).

The remainder of the file contains event records and these are used for analysis. The record definition appears in Table 3-1. The event records are subdivided into groups based on channel number. The events records are in chronological order. User records are defined as all those records for a channel which are between a channel ringing record and the first channel disconnect record inclusive. Specifically, the channel ringing is when the low-order byte of US.FLG** equals 5, and the channel disconnect is when the low-order byte of US.FLG equals 6 (Tables 3-2, 3-3). The statistics for each user is put on one line of the report. However, any user whose second event is channel disconnect is not recorded in the statistics.

Using the user data, the gross statistics are computed. The time of day in column 1 is the GMT time on the event record channel disconnect. The briefing time in column 6 is the difference in time between the time of day on the channel ringing event and the channel disconnect event.

In order to get the counts in columns 2 through 5, each user's records are divided into passes. A pass consists of 2 phases, (1) a location identifier entering phase and (2) a report requesting phase.

TABLE 3-1. FILE VRDATA.DAT RECORD FORMAT

Word	Byte	Contents
1		Recorder Header (contains the value -16)
2	3	Length in bytes of the variable data record
3	5	Channel Number being recorded (U.S. CHN)
4		Line Status (US.STA)
5	10	Current state of the protocol (US.KEY)
6	11,12	User's Current Status (US.FLG)
7	13,14	Additional status information (US.PER)
8	15,16	Low-Order Time since midnight in seconds
9	17,20	High-Order Time since midnight
10+	21+	Variable data record (length given in word 2)

The location identifier count in column 2 is done in two parts. First, the location identifier entering phase is checked to locate the record which has the greatest location identifier count. This becomes the location identifier count for that pass through the briefing. Then the location identifier counts for each pass are summed up for the user to arrive at the total location identifier count for the user in column 2. The location identifier input phase is defined by the high-order byte of US.KEY being either equal to 1 (EOCID) or 5 (NXTLOC).

The report type count is computed for each briefing pass by checking the high-order byte of US.KEY. If a report is requested during that pass (i.e., US.KEY byte value equals 15 for SA's, 17 for FT's, 20 for GF's) the number of reports of that type requested for that pass equals the number of location identifiers entered that pass. This is then summed over the passes for that user and entered in columns 3, 4 and 5. Use of the REPEAT function may lead to getting multiple report counts in 1 pass.

1. Twenty Channel Voice Response System, Input Output Computer Services, Inc., Final Report.

*Section 2.3.2, contains a description of the statistics file (VRDATA.DAT).

Appendix B also provides a partial dump of file VRDATA.DAT.

**Refer to Tables 3-2 and 3-3 for definition of U.S.KEY and US.FLG values respectively.

TABLE 3-2. US.KEY (HIGH-ORDER BYTE)

<u>Value (Octal)</u>	<u>Flag Word</u>
0	HELLO
1	LOCID
2	BRIEF
3	WHATBR
4	BRIEF2
5	NXTLOC
6	POUND
7	COMPLI
10	RPTYP
11	HOURS
12	ALTI
13	MORE
14	NOTAM
15	SUROVB (SA's)
16	PIREPS
17	TERMFC (FT's)
20	WINDS (GF's)
21	PHOURS
22	PALTI
23	PSYN
24	WARN
25	LATOB
26	FORCAS
27	WINDAL
30	EHOURL
31	EALTI
32	ENROUT
33	ENALT
34	ENETA

TABLE 3-3. US.FLG (LOW-ORDER BYTE)

<u>Value (Octal)</u>	
1	# sign indicator
2	* indicator
3	Talk mode indicator
4	Invalid keystroke
5	Channel ringing indicator
6	Channel disconnect indicator
7	YES response
10	NO response
11	Return to interrupted code
12	Briefing return
13	Repeat last message unit
14	Cancel last input
15	Proceed with speaking
16	Stop present activity.

TABLE 3-3. US.FLG (HIGH-ORDER BYTE) (CONTINUED)

<u>Value (Octal)</u>	<u>Meaning</u>
1	Enable Data Set
2	Numeric Input Expected
4	Cycle to Next DAP Value
10	Echo Input Flag
20	Phonetic Echo
40	Do Not Enable Data Set
100	Speak Done Indicator
200	Echo Done Indicator

The allocation of time to overhead, location, SA, PT and GF are allocated by use of the high-order byte of US.KEY. When 2 consecutive records for 1 user have different US.KEY values the time between them is allocated to the second record US.KEY value. For this purpose: (1) US.KEY values 1 (LOCID) and 5 (NXTLOC) are considered equal; (2) US.KEY values 20 (GF), 21 (PHOURS) and 22 (PALT1) are GF reports, and (3) US.KEY values 2 (BRIEF) and 4 (BRIEF2) are assigned to the report request preceding them.

An example of assigning US.KEY values BRIEF and BRIEF2. If there are 3 consecutive records for 1 user and the first US.KEY value is 15 (SA), the second record 2 (BRIEF) and the third record 4 (BRIEF2), the routine assumes all US.KEY values are 15 (SA).

Columns 11 through 15 contain counts of the special functions. As each record is processed, the lower-order byte of US.STA is checked (Table 3-4). If the byte has a value from 42 to 50 octal, it is processed by incrementing the counter for that special function.

TABLE 3-4. SPECIAL FUNCTION CODES (US.STA - LOW-ORDER BYTE)

<u>Function</u>	<u>Octal Value</u>	<u>Explanation</u>
Stop	42	Briefing stopped by user
Go	43	Briefing restarted by user
Begin	44	Recycle to beginning of protocol
Repeat	45	Briefing repeated by user
Jump	46	Report skipped by user
Delete	50	Cancel last entry

4. PROGRAM STRUCTURE

The routines used in this VRS statistics program may be divided into 7 groups (see Appendix C). The first group consists only of the main program which has 2 functions: (1) to screen out data based on channel number and option,* and (2) to call the other routines of the program.

The second group is the read group. This program consists of all routines involved in reading the trace file. This group opens the file, locates the starting block and byte, provides 1 record at a time on request and closes the file when the end of file is reached.

The third group is the user group. This group looks for records with channel ringing and channel disconnect status. When a channel ringing record is found, it records the starting time. When a channel disconnect status is found, it stores the time of day, and the connect time. It screens out those users who have only a channel ring-up and a channel disconnect event.

As part of the user group, the count of users for each hour is stored and the maximum number of users per hour is stored.

The fourth group is the locations group. This group looks for records with LOCID and NXTLOC status. It counts for each user the number of location identifiers requested.

Under option 2, it will also store the actual 3 letter location identifier codes requested by the user. Under option 4, it will store the location identifiers and the number of requests for that location for all users for the entire data collection period.

The fifth group is the status group. It checks all records and counts the number of records having a given status.

The sixth group is the time group. This group records the time between 2 specified events. It is done both for individual users and summed up for the day.

The seventh group is the output group. Under option 1, a line is printed for each user. The header and daily summary are always printed.

*Statistical report options are described in Section 5.

TITLE: STATA (Figure 4-1)

STATA is the main program and has 3 functions. The first is to read in a record and make sure the user number of the record is between 1 and 20, inclusive.

The second is to store a copy of the current record and the previous record for each user.

The third function is to call the various processing programs in order. The routines directly called are:

1. READER: Reads in a record.
2. MINUTE: Converts time in seconds on the record to minutes and tenths of minutes.
3. LOCO: Counts location identifiers.
4. MORE: Saves data from all previous passes when there are multiple passes.
5. UMAX: Computes the maximum number of simultaneous users for each hour and for the day.
6. REPORT: Computes the number of weather reports, by type, for each user.
7. STATUS: Counts the number of times each special function is used for each user.
8. TIMER: Computes the user connect time, and the time of day for each user. It also types a line for each user. As part of this routine, an hourly count of users and their total connect time are accumulated.
9. SUMMARY: Computes the total number of users for the day and the average connect time. It then prints the daily summary.
10. COUNT: Computes a matrix counting the number of records. The columns are DAP KEYS from US.KEY and the rows are flags from the low-order of US.FLG.
11. KEY: Counts the total number of users and the total number of users who hang up before requesting a report.

The routines not directly called are:

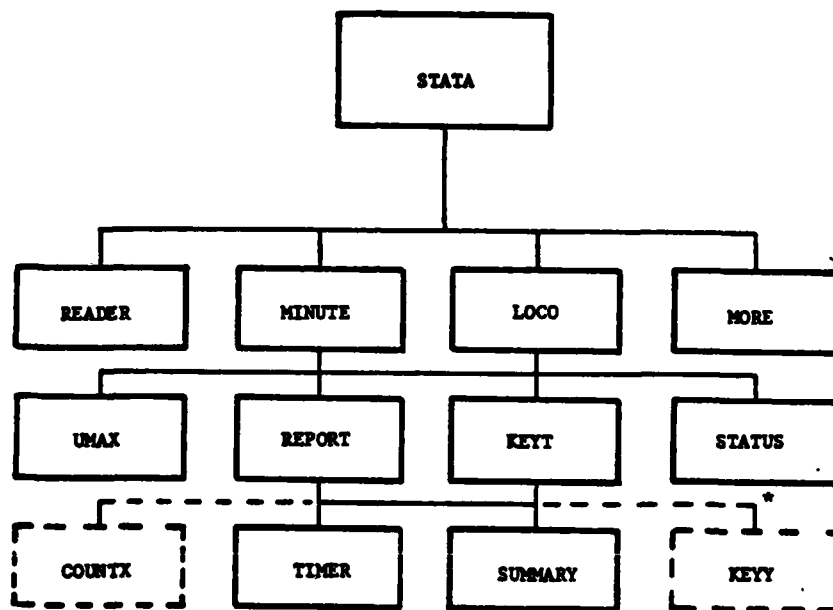
12. OPENEN
13. INIT
14. CLOSER
15. READS
16. START
17. FINISH
18. BATCH CHECK
19. TYPEN
20. LOCAT
21. LOCU
22. ERROR

TITLE: STATA

Read a record using routine READER.

Is there an end of file?

Then:	Else:	
Call routine summary	Is the user number	between 1-20
STOP	Then:	Else:
	Call MINUTE	Go to beg.
	Call LOCO	
	Call MORE	
	Call UMAX	
	Call REPORT	
	Call KEYT	
	Call STATUS	
	Call TIMER	
	Go to beginning	



* - - - - - Compatible routines not normally used.

FIGURE 4-1. MAIN PROGRAM AND DIRECTLY CALLED PROGRAMS

TITLE: READER (Figure 4-2)

READER is the routine which handles the interface with the data file and provides the data 1 record at a time to routine STATA. On the first request for a record, the file is opened using routine OPENER.

Then routine INIT reads in block 0. In block 0 is the starting point and the ending point for trace data. Routine INIT passes on to READER the last block and the last word, which will be used to set the end of file check. It uses routine READS to read in the first block of valid data and sets the word pointer to the first data word.

In all read requests, the program reads in a record from the buffer. Since the records have variable lengths, this must be done in 2 steps. In step 1, 4 bytes are read in. Then routine ERROR checks to insure the record is valid. If it is not valid, then the ERROR routine sets the word pointer and returns. If there is no error, then the record length is set to 14 plus the contents of byte 3 and the remainder of the record is read in from the buffer using routine READS. It is the remainder of the record which is transferred to routine STATA.

TITLE: READER

Is this block zero?

Then:

Call OPENER
Call INIT

Else:

BEG

Set byte count to 4
Call READS
Call ERROR

Is there a reading error?

Then:

Return to BEG

Else:

Word count = 14+ contents byte 3
Call READS
Is this an end of file?

Then:

Set EOF
INDICATOR
Call CLOSER

RETURN

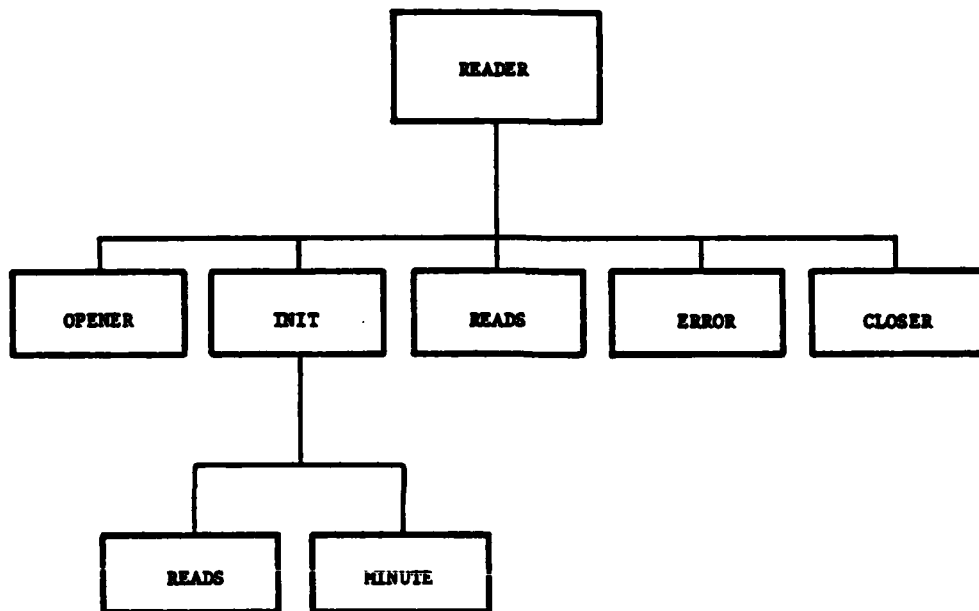


FIGURE 4-2. PROGRAMS USED TO READ THE RAW DATA FILE

TITLE: MINUTE

In each record, words 8 and 9 contain the time in seconds. This requires 17 bits. Word 8 contains the lower 16 bits and word 9 contains the seventeenth bit as bit 0 of the ninth word. The program shifts the bits 2 positions right (division by 4) and then multiplies the result by 4/60 to convert to minutes. The result is placed in the word TIME.

TITLE: MINUTE

Cur6 - Current record word 8

Cur7 - Current word 9

TIME - Time in minutes; set to 0

Is cur7 bit zero set to 1?

Then:

Set bit 12 of TIME to 1

Else:

Do while, I goes from 6 to 15?

Is cur6 bit I set to 1?

Then:

Set TIME bit (I-5) to 1

Else:

RETURN

This routine counts the number of location identifiers entered by the user. The number of locations count is not recorded in the raw statistics directly and so the following method was chosen. Each record contains the last ASCII message sent to the PDP 11/70. Therefore, a search will be made for the ASCII messages which contain the location identifiers and a count will be made.

The search begins 2 bytes beyond the "P". The first test is that there be 3 letters (a possible location). The count of locations is incremented by 1 if the test is passed. The second test is to check if the 3 letters are followed by a slash. All locations except the last on this message must have a slash following the location code. Thus, the location count is terminated when the slash test fails.

TITLE: LOCO

Is IRE(16) equal to 5 or 1?

Go to 77

Else:

Else:

Eligible:

13

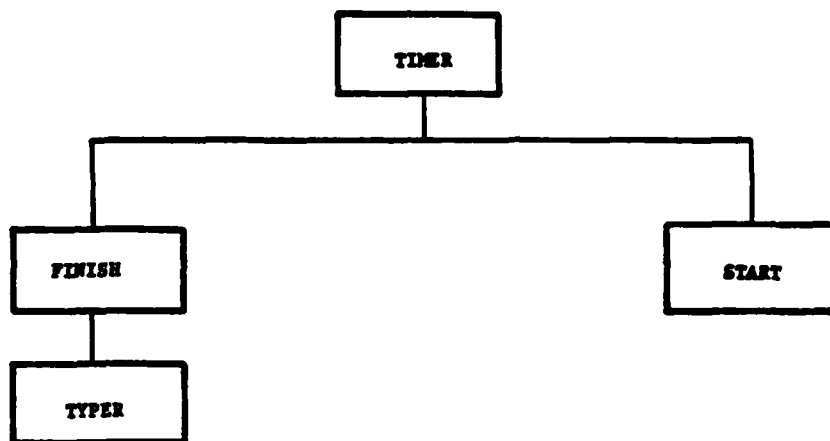


FIGURE 4-3. ALL ROUTINES WHICH DETERMINE WHEN A USER HAS LOGGED ON AN OFF

TITLE: MORE

Routine MORE produces 3 location requested counts: (1) locations requested by each user, (2) locations requested by each user for each briefing pass, and (3) locations requested by each user during the current pass. A pass consists of an input of locations followed by a request for reports.

The inputs to this program are a user number, a location count, the high-order byte of US.KEY and the lower-order byte of US.FLG. The output is the total number of locations requested by each user for the output routine TYPED, and the number of locations for this pass to the report counter routine REPORT.

When a record is to be processed, the first thing to be tested is the US.FLG byte. If it indicates a new user (value equals 5), then all counts are set to 0 and the new pass indicator is set to 0. The routine is then exited.

Next the US.KEY byte is checked. If it indicates reports are being requested (value greater than 5), the total locations requested is set equal to the sum of the total number of locations entered in previous passes plus the number of locations entered in this pass. The new pass indicator is set equal to 1 and the routine exited.

Next the US.KEY byte is checked for 1 (LOCID) or 5 (NXTLOC). If it is neither of these, the routine exits. If it is either of these, the new pass indicator is checked.

If the new pass indicator is 0, the count for this record is compared to the count for this pass and the larger value is entered in the count for this pass. Then the routine is exited.

If the new pass indicator is 1 then a new pass is starting. Hence the count for current pass is added to the count for previous passes, the count for this pass is set to the count for this record and the new pass indicator is set to 0. Then the routine is exited.

TITLE: MORE

IUSER - User channel number

COUNT - Count of locations, latest count for user

COUNTL - Maximum count for this pass and this user

COUNTM - Sum of maximum counts for all previous passes

Has US.KEY returned to 5 (NXTLOC)

Then:

COUNTL = COUNTL - 1

COUNTM

COUNTL = 0

Else:

RETURN

TITLE: UMAX

Routine UMAX computes the maximum number of simultaneous users for each hour. The inputs are the user, the time, and the low-order byte of US.FLG.

When a record has a US.FLG value (low-order byte) of 5, a check is made of the user channel active indicator. If the indicator is clear, the user count is incremented by 1.

When a record has a US.FLG value of 6, a check is made of the user channel active indicator. If the indicator is set, the user count is decremented by 1 and the user channel active indicator is cleared.

In all cases, the current count is compared with the maximum count for the hour and the maximum count is adjusted if required.

TITLE: REPORT

This routine counts the number of reports by type and user.

If a record has a 5 in the low-order byte of US.FLG (channel ringing), the counts for that user is set to 0.

If the US.KEY high-order byte indicates a report type, the count for that report type and that user is incremented by the location count for that user and the current pass.

TITLE: REPORT

COUNTL Current pass location count

IUSER User

SAI SA count for each user

FTI FT count for each user

FDI GF count for each user

Is US.FLG low-order byte 5?

Then:

SAI (IUSER) = 0

FTI (IUSER) = 0

FDI (IUSER) = 0

Else:

Is US.KEY high-order byte = 15

Then:

INCREMENT SAI

Else:

Is US.KEY = 17

Then:

INCREMENT FTI

Else

Is US.KEY = 20

Then:

INCREMENT
FDI

Else:

RETURN

TITLE: STATUS

Routine STATUS checks the status byte to determine the status command requested by the user. A count of commands are collected and as part of the output there is a list of status commands and how often they were requested.

TITLE: TIMER (Figure 4-4)

Routine TIMER is used to decide whether a user is to be counted in the statistics. Currently, there are 2 criteria: (1) the user must have a channel ringing and a channel disconnect event, and (2) the user must have 1 other event.

Every time a channel ringing event occurs, routine START is called to initialize the channel.

Every time a channel disconnect is encountered, a check is made as whether to count this user. Current criteria are: (1) the channel must be active, and (2) there must be at least 1 event between the channel ringing and channel disconnect events. If the criteria are met, then routine FINISH is called.

Routine START sets the user start time to the current time and sets the channel active indicator.

Routine FINISH computes time active as current time minus start time. It then calls routine TYPED to print a line.

TITLE: TIMER

Is low-order byte of US.FLG equal to 5?

Then:

Call START

Else:

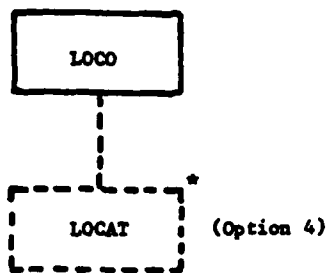
Is US.FLG equal to 6?

Then:

Call FINISH

Else:

RETURN



*
----- Not normally used.

FIGURE 4-4. LOCATION-ORIENTED OPTIONS

TITLE: SUMMARY

This routine is called by the main program when file processing is complete. It prints hourly statistics, followed by daily statistics. The hourly statistics are the number of users, the average time a user is on the line, and the maximum number of simultaneous users. The daily statistics are total number of users, the average user time, and the maximum number of simultaneous users. Further there is a breakdown of how the user time is used.

When option 4 is used, a listing of the location identifiers and times called is produced.

TITLE: COUNTX

This routine sets up a 2 dimensional count array where 1 subscript is the high-order byte of US.KEY and the other subscript is the lower-byte of US.FLG. When a record is read, the US.FLG byte is checked. If the value is 5 and the last value of US.FLG for that user is 5 or 6, the record is not counted. If the value is 5 and the last US.FLG for that user was 5 the record is not counted. In any other case, the count array is incremented by 1.

After doing the count, the current value of the US.FLG byte is stored in the flag array for that user and the type routine is executed.

The type routine provides for a typing of all non-zero elements of the count array at the end of each hour of VRS operations.

TITLE: COUNTX

INPUT: Cur6 US.KEY high-order byte
 Cur7 US.FLG low-order byte
 Time VRS Time
 IUSER User channel

Is Cur7 and FLG (IUSER) both 6?

Then:	Else:		
	Is Cur7 equal to 6 and FLG (IUSER) equal to 5?		
	Then:	Else:	
		Is Cur7 equal 5 and FLG (IUSER) equal 5?	
		Then:	Else:
			Increment Counter (Cur6, Cur7)

FLG (IUSER) = Cur7

RETURN

TITLE: KEYY

This routine is a diagnostic tool which reads each event record and makes event counts (Table 4-1). The terms used are: active (having a user on the channel) and inactive (having no user on the channel).

First each event is checked to determine whether the event is activation of a channel or the deactivation of a channel. If it is either, a counter is incremented.

Next the channel's prior status is checked to see if it was active. If it was not, then the following checks are made. First, a check is made to determine whether the channel is cleared. If it is not, counter 7 is incremented. Second, is the channel being activated? If it is, then counter one is incremented.

If the channel was previously activated, the following checks are made. Is the active channel being activated? If it is, then counter four is incremented. Next a check is made to determine whether the channel is being deactivated. If it is being deactivated, the following checks are made. Were any locations entered? If not, then counter five is incremented. Were any reports requested? If not, then counter 2 is incremented.

TABLE 4-1. TEST TABLE

Counter	Current Event	Status	Previous Event	Key*
1	activation	inactive	-	IREC7 = 5
2	deactivation	active	-	KEYM = 5
3	activation	-	-	IREC7 = 5
4	activation	motive	-	IREC7 = 5
5	deactivation	motive	activation	JREC = 5
6	inactivation	-	-	IREC7 = 6
7	inactivation	channel not cleared	-	IREC6 = 0
IREC7 IREC6 JREC KEYM		Low-order byte, US.FLG High-order byte, US.KEY Previous value of IREC7 Maximum value IREC6		

*Key is for each user.

TITLE: OPENER

This routine calls on library routines to open the disc file. If in any step an error occurs or there is no room for this input, the program stops.

Routine ICHAN allocates a channel.

Routine LOOKUP opens the file, allocates buffer space and sets the buffer pointer at the start of the file.

TITLE: INIT

This routine is called routine READER after the data file is open. Using routine READS it reads in block 0. It then decodes the control data and types it out. In word 1 is the date when the file was initiated. In word 2 is the low-order bits of time and in word 3 the high-order time bits. In word 4 is the starting block number (usually block 1) and in word 5 the specific starting byte offset (usually 0). The next 5 words have the data for the file end. Word 6 has the date, word 7 and word 8 the time, word 9 the block number and word 10 the byte offset.

Under option 8 the user may input a new starting block and a new starting byte offset. Inputting block 1, byte 0 is the same as option 0.

TITLE: INIT

Extract bits from the word 1:

Bits (4-0) plus 72 equal the year

Bits (9-5) equal the day of month

Bits (14-10) equal the month

Extract bits from word 6:

Bits (4-0) plus 72 equal year

Bits (9-5) equal day of month

Bits (14-10) equal the month

Convert time in words 2 and 3 (time in seconds) to time in minutes using routine MINUTES.

Print results.

Is this option 8?

Then:

Input block and byte offset from
Terminal in (I4) format.

Else

RETURN

TITLE: CLOSER

This routine calls on library routines to close and free the channel.

CLOSE closes the channel.

IFREEC frees the channel.

TITLE: READS

This routine manages the buffer filling and provides records or parts of records as required. The data on the disc are in blocks of 516 bytes. For initialization the buffer pointer is set at 512 and the block counter at 0.

The routine is called to provide a specified number of bytes. If based on the buffer pointer, there are enough bytes remaining in the buffer, the bytes of data are transferred from the buffer to the specified array. Then the buffer pointer is incremented by the number of bytes transferred. Then the routine exits.

If there are two few bytes left in the buffer, then the data in the buffer are transferred to the array and a record made of bytes transferred. Next library routine READW is used to read in a block of 512 bytes. The block count is increment, the buffer pointer is set to the beginning of the buffer and a second data transfer is executed.

TITLE: START

Routine START is called by routine TIMER whenever a channel ringing event occurs. The routine sets the user start time to the time on the event record, and sets the active channel indicator.

Set the start time to the time on the record. Set the active channel indicator.

Return.

TITLE: FINISH

Routine FINISH computes the time a user has been on the system, updates summary statistics, and arranges for the printing of a line for each user.

The first step is to check if there is a bonafid user. This means that the active user for this channel is set and the previous record for this channel did not have a 5 in the low-order byte of US.FLG.

If the record fails the test, the routine is exited. If the test is passed, the time of day is set to the time on the record, the time for this user is computed as the time on the record minus the start time for this channel, the cumulative statistics are updated, and a line is printed.

The cumulative statistics are the number of users for the hour in this record and the total time users are on during the hour in this record. The user count is incremented by 1, while the total time for the hour is incremented by the time of this user.

TITLE: FINISH

Is the active user indicator for this channel set?

Then:

Was the previous US.FLG equal to 5?

Then:

Else:

Time = record time -
start time

Total Time = Total time
+ time

User Count = User count
+ 1

Call TYPED

Else:

TITLE: BREAK CHECK

This routine is called by timer to determine if there was a break in the data input. A break occurs when the time of the last input is more than 60 minutes before the time of the current input. This is done by comparing the input time with the previous input LTIME. In this comparison, special care must be made for the transition from 2300 GMT to 0 GMT.

If a break is found, then all active users are terminated and the counters and indicators for the last hour are reinitiated.

TITLE: BREAK CHECK

TIME Current time
LTIME Previous time
KTIME Time difference

$$KTIME = TIME - LTIME$$

Is time less than 60 and KTIME less than 0?

Then:

KTIME = KTIME + 1440

Else:

Is KTIME greater than 60?

Then:

ST = -1
KEY = 3
IND = 3

Else:

RETURN

TITLE: TYPER

This routine has data prepared by other routines and stored in global commons. When this routine is called, it types one line based on the type statement and the format. Currently, it types the time of day in minutes, the user briefing time, the user count for the hours, the running average briefing time for the hours, the user number, the number of locations requested by the user and the report types requested by the user.

Type line

RETURN

TITLE: LOCAT

Routine LOCAT is called by routine LOCO. The input is a 3 character location identifier. LOCAT then prepares a list of all the location identifiers inputted by all the users. It also keeps a count of the number of requests for data on each location.

TITLE: LOCAT

NLOC, Number of location identifiers; set NLOC = 0

Read in 3 letter code.

Is NLOC greater than 0?

Then:

Is the entry a new location identifier?

Else:

NLOC = 1

Then:

NLOC = NLOC + 1

Enter Number into table

Else:

Enter Location Identifier in table

RETURN

TITLE: LOCU

This routine sets up an array for each channel. Every time routine LOCO checks a location identification and determines the location it has 3 letters. LOCO calls LOCU which stores the name in that users array. This array is printed every time a line is hung up. Routine TYPED does the printing of the array contained in common ENTS.

TITLE: LOCU

ENT4, number of locations stored for user; set ENT4 = 0.

Read in 3 letter code.

Increment ENT4 for user.

Does ENT4 for user exceed 10?

Then:

ENT4 for user = 10

Place first letter in ENT (1, ENT4)

Place second letter in ENT (2, ENT4)

Place third letter in ENT (3, ENT4)

Else:

RETURN

TITLE: ERROR

Routine error checks the first 2 bytes of each new record. If it is the legitimate record start (byte 1 equals minus 1) it allows processing. If it is not legitimate the file is searched for the first 2 bytes which indicate the record is legitimate.

5. STATISTICS PROGRAM OPERATION

This section contains the procedures for building and running the VRS statistics program. The program requires that the raw data file be named TRACE.DAT. Hence before running the statistics program, rename or copy the data file of interest to TRACE.DAT.

Program Compilation and Task Building (RTL1-Version 2)

A. Program Compilation

```
.R FORTRAN
STATQ = STATQ
STATC = STATC
STATD = STATD
```

B. TASK BUILD COMMAND

```
.R LINK
STATA = STATQ, STATC, STATD, SYSF4, FORL1B
```

C. OPERATING INSTRUCTIONS

1. Enter on terminal:
 _R STATA (Run Program STATA)
2. Enter 2-digit option number (Table 5-1).
3. For options 8 through 15, type block and byte numbers, each in I4 format (e.g., for block 100, byte 52: 0100 0052).
4. Enter a 2 digit minimum channel number between 01 and 20.
5. Enter a 2 digit maximum channel number between 01 and 20.
6. The program then outputs the statistical report according to the options selected by the user.

TABLE 5-1. OPTION TABLE

Option*	Action
0	Print summary only
1	Print a line for each user
2	Store location codes for each user
4	Print location ID and demand
8	Manual control of starting block

*A combination of options can be requested by adding option codes.

APPENDIX A - SAMPLE VRDATA.DAT FILE (PARTIAL DUMP - 1ST 3 BLOCKS)

DKU:TRACF.DAT

BLOCK NUMBER 00000 (*Header Block*)

000/	004710	164354	000000	000001	000000	004750	113601	000000	*H.LH.....H.....*
020/	000354	000406	000000	000000	000000	000000	000000	000000	*W.....*
040/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
060/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
100/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
120/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
140/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
160/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
200/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
220/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
240/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
260/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
300/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
320/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
340/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
360/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
400/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
420/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
440/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
460/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
500/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
520/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
540/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
560/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
600/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
620/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
640/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
660/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
700/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
720/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
740/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*
760/	000000	000000	000000	000000	000000	000000	000000	000000	*.....*

BLOCK NUMBER 00002

000/	000000	165026	000000	000063	177762	000014	030401	000011	*...J...3...1...*
020/	002400	050411	000402	165032	000000	050061	020515	051117	*...U...J...1PM-5P*
040/	020104	023457	000015	177762	000010	030401	000011	000400	*D /'...F...J...*
060/	040411	000402	165040	000000	051461	022501	021446	000015	*A... J...1SAR...*
100/	177762	000014	030401	000041	007400	040006	000412	165103	*R...1!...CJ*
120/	000000	030446	030102	030060	036063	023452	000015	177762	*...180003C*...*
140/	000014	030401	000041	007400	040011	001412	165105	000000	*...1!...EJ...*
160/	030446	030101	030060	035463	023452	000015	177762	000002	*61A0003?*'...*
200/	030401	000041	000000	040011	000000	165106	000000	000061	*.1!...FJ...1.*
220/	177762	000010	031002	000011	007400	040411	000402	165114	*F...2...A...LJ*
240/	000000	043062	026124	021446	000015	177762	000020	031403	*...2FT...P...3*
260/	000011	002400	050411	000402	165120	000000	050063	020515	*...L...FJ...3P...*
300/	044120	027530	043104	036527	025467	000015	177762	000016	*PMX/DFW=7+...*
320/	031403	000011	011000	045411	000402	165117	000000	043063	*.3...K...J...3P*
340/	026504	026462	034463	022460	024457	000015	177762	000002	*0-2-1508/)...*
360/	030401	000040	000000	040005	000000	165241	000000	000061	*.1...J...1.*
400/	177762	000014	030000	000041	010000	040006	000412	165242	*P...0!...J*
420/	000000	030046	030102	030460	034460	023452	000015	177762	*...60000104*...*
440/	000014	030000	000041	010000	040011	001412	165244	000000	*...0!...J...*
460/	030046	030101	030460	034060	023452	000015	177762	000002	*60A0010H*...F...*
500/	030000	000041	000000	040011	000000	165244	000000	000060	*.0!...J...J...*
520/	177762	000014	031002	000011	011000	045411	000402	165272	*R...2...J*
540/	000000	043062	026504	032062	031455	037460	024456	000015	*...2FD-24-30?...)...
560/	177762	000002	033006	000040	000000	040005	000000	165276	*R...6...J*
600/	000000	000066	177762	000002	033407	000040	000000	040005	*...F...7...*
620/	000000	165276	000000	000067	177762	000014	031403	000041	*...J...7...J!...
640/	005400	040006	000412	165304	000000	031446	030102	030060	*...4...J...63F000*
660/	020065	023453	000015	177762	000014	031403	000041	005400	*5 +'.K...3!...*
700/	040011	001412	165306	000000	031446	030101	030060	037465	*...FJ...63A0005?*
720/	023452	000015	177762	000002	031403	000041	000000	040011	*...F...3!...*
740/	000000	165306	000000	000063	177762	000020	030401	000011	*...FJ...3...1...*
760/	002400	050411	000402	165307	000000	050061	020515	050523	*...G...GJ...1PM-SY*

IDENTITY CODES AND REQUEST COUNTS

LOC COUNT	LOC COUNT	LOC COUNT	LOC COUNT	LOC COUNT	LOC COUNT
IAU 60	UCA 130	ABD 11	SAN 5	VRR 6	TPA 15
ORD 14	PHX 10	DFW 7	SYR 2	CMH 124	USU 12
MIA 38	HAI 36	HGR 6	PJC 14	CLE 12	MTS 18
EVV 3	STL 17	MKC 4	DDC 3	MAR 13	HFD 1
JFK 53	PSP 12	RIS 2	MLA 1	ROU 12	AGC 11
MDT 6	ADW 8	SLC 4	DAY 17	FRA 10	TOL 7
MIE 2	PHL 4	HPR 1	CVG 6	ZZV 5	LUN 4
CAK 4	REL 1	MHT 2	ACU 5	GJT 1	GUC 2
JAC 1	PIT 11	ROS 13	MFD 18	MTT 1	PRI 1
SAV 7	JST 1	AUU 2	DFL 4	JAX 13	LAX 47
PSP 5	ICT 16	GCK 2	ALS 1	FNN 1	FNC 1
RLH 1	CHS 4	IND 12	PCF 1	FLO 2	DAB 5
TYS 1	OHF 8	PYR 8	FAY 1	PUB 1	SLN 1
ENR 4	IBL 1	IPZ 2	FAA 10	LOH 1	AMA 3
OKC 8	PSK 1	ALP 3	CXY 7	FLL 4	FDY 10
BGM 1	CFW 10	FSM 16	SPA 1	RSA 1	FEF 3
ILG 1	RDG 1	ATL 10	GHI 3	MRR 7	CUC 3
PSC 4	MCP 1	PHF 1	AOS 1	PVC 2	GFY 14
SFO 5	RAP 15	MSN 15	GLO 2	DEN 27	CHU 2
CMA 1	ILM 2	CFE 1	YRG 4	CRF 3	CON 1
NWZ 1	HCA 6	CHA 7	DAL 14	OMA 4	HIT 1
ERI 1	BLF 3	PGI 1	AGK 3	UAA 1	LAS 2
CKR 4	PKB 3	MOB 2	SRA 1	ECF 1	KCI 2
COS 4	DLW 3	SHN 4	AZO 1	MSY 5	PIA 4
DSM 3	SNY 1	HDA 4	LIT 2	JAN 4	KXF 3
HGF 3	TLH 3	RTI 3	BIA 3	FSD 2	LGH 1
ATY 1	CWI 1	FAH 2	AVL 1	SPI 1	KRO 2
SCK 1	TGP 1	GGO 1	APF 1	BFF 1	GAF 1
DUI 3	SHD 1	SSU 1	NVH 3	GHV 6	SSY 1
FNY 1	TRI 1	MOU 2	RCA 1	UFF 1	DEC 1
SBY 3	ANW 1	PDW 2	PUC 3	ETV 4	JFC 4
HOW 1	DAA 1	APL 3	TVC 2	HCA 1	ASE 2
RUP 2	BJT 15	GEZ 1	CKH 1	MHT 1	CLT 2
GSC 1	HCA 1	MDA 1	MPV 1	LGA 9	KNU 2
TEB 2	TUL 2	MBS 1	SAC 4	AVF 1	KNO 2
GHE 1	GFA 1	YYY 1	CAE 2	AGS 1	MLI 1
SLA 5	CID 1	ELM 1	DFT 1	DHN 1	IAM 3
MGL 1	ZZZ 4	VNY 4	TYN 1	RZN 3	ING 1
BOI 1	ZIH 1	LAV 5	LAG 23	CAN 1	JFF 31
MNG 2	DEF 9	GHJ 9	JKL 9	JKK 9	UGG 7
RIA 2	IFL 1	FLP 1	BFL 3	FAT 2	HLV 1
INT 1	PLF 1	CRF 1	HLG 1	HYG 1	GAX 1
ONT 1	OPR 1				

HOOR	USERS	AVG. TIME	SIMUL. USERS
16	13	3.700	6
17	46	4.500	7
18	34	3.500	6
19	59	3.600	8
20	44	4.300	7
21	52	2.900	10
22	43	4.000	6
23	21	3.000	4
0	5	3.800	5
1	14	3.400	5
2	25	4.600	7
3	21	3.500	5
4	21	4.000	4
5	15	6.100	6
6	6	6.200	3
7	5	9.100	2
8	1	5.800	1
9	3	3.700	1
10	4	3.100	3

TOTAL USERS 447 AVG. TIME 3.988 MAX. SIMUL. USERS 10 FOR DAY

FRACTIONAL %
 OVHD 21
 LOCID 11
 SA 27
 FT 31
 GF 10

BLOCK NUMBER 00001 (user data blocks)

000/	177762	000002	030000	000040	000000	040005	000000	164434	*R.....0.....0.....1*
020/	000000	000000	177762	000014	030000	000011	002400	050411	*...C...F.....0.....0*
040/	000402	164455	000000	050060	026515	040511	024104	023456	*...-I...CPM-140(....*
060/	000015	177762	000010	030000	000011	006400	040411	000402	*...K.....0.....A...*
100/	164463	000000	051460	022101	021446	000015	177762	000014	*31...CSASAS...R...*
120/	030000	000041	003400	002406	000412	164533	000000	030046	*.G!.....11...60*
140/	030102	030060	035463	023452	000015	177762	000014	030000	*00003;*.R.....0*
160/	000041	003400	002411	001412	164535	000000	030046	030101	*!.....11...60A0*
200/	030000	035063	023452	000015	177762	000002	030000	000041	*003;*.R.....0!..*
220/	000000	042411	000000	164535	000000	000000	177762	000002	*...E..11...0.P...*
240/	030000	000041	000000	042411	000000	164540	000000	000060	*.0!.....E...1...0.*
260/	177762	000002	030000	000041	000000	042411	000000	164540	*R.....0!.....E...61*
300/	000000	000060	177762	000002	030000	000000	000000	042405	*...C.P.....0.....E*
320/	000000	164574	000000	000060	177762	000024	030000	000011	*...1...0.K.....0...*
340/	002400	050411	000402	164645	000000	050060	026515	041504	*...G...A1...UPM-DC*
360/	027501	041101	027521	040523	033111	027476	000015	177762	*A/AMU/SAN63/...U.*
400/	000010	030000	000011	006400	040411	000402	164653	000000	*...G.....A...+1...*
420/	051460	022101	021446	000015	177762	000002	030401	000040	*USASAS...W.....1...*
440/	000000	040005	000000	164701	000000	000061	177762	000002	*...A...A1...1.P...*
460/	031002	000040	000000	040005	000000	164723	000000	000062	*.7.....W...SI...2.*
500/	177762	000006	030401	000041	000400	054006	000012	164741	*R.....1!.....X...4!*
520/	000000	050061	026515	041101	177762	000014	030401	000041	*...1PM-APK.....1!..*
540/	000400	054011	001012	164743	000000	030446	030101	030060	*...X...CI...61A000*
560/	034060	023452	000015	177762	000002	030401	000041	000000	*04*...R.....1!...*
600/	054011	000000	164743	000000	000061	177762	000010	030000	*.XJ.CI...1.W.....0*
620/	000011	007400	040411	000402	164775	000000	043060	025124	*.....A...11...0P1*
640/	021446	000015	177762	000020	031002	000011	002400	050411	*6...R.....2.....U*
660/	000402	164775	000000	050062	026515	051126	027502	050124	*...11...2PM-VPM/TP*
700/	035101	025467	000015	177762	000010	031002	000011	006400	*A:7+...W.....2....*
720/	040411	000402	165004	000000	051462	023101	021446	000015	*.A...J...2SA66*...*
740/	177762	000002	030401	000040	000000	054005	000000	165007	*W.....1.....X.....*
760/	000000	000061	177762	000002	031403	000040	000000	040005	*...1.k.....3.....*

* Typical user record

WMS STATISTICS

VMS STATISTICS					
MINIMUM CHANNEL		MAXIMUM CHANNEL		VMS	
START	DATE	START	DATE	1 BYTE	0 TIME
20	2 14	20	2 14	238	267
20	2 15	20	2 15	238	267
20	2 15	20	2 15	238	267

GMT	TIME	COUNT		FL	GS	TIME	INFR	TIMES (0.1 MIN)			STOP	COUNT SPFC. PUCT			JUP	NPG	LOC ID	VFOURSID
		LUC	SA					LUC	SA	FT		GO	MPI	DEL				
16125		1	1	0	0	10	2	1	7	0	0	0	0	0	0	0	IAP	
16135		1	1	0	0	10	4	1	5	0	0	0	0	0	0	0	UND	
16139		3	3	3	0	46	6	1	14	27	0	0	0	0	0	0	UCA	AMQ SAN
16141		2	0	0	2	29	10	6	0	0	13	0	0	0	0	0	PWX	HFH
16144		2	2	2	0	27	5	1	10	11	0	0	0	0	0	0	CWM	USU
16146		2	2	0	39	6	2	10	21	0	0	0	0	0	0	0	SVS	DCA
16145		3	3	3	3	75	7	2	18	24	20	0	0	0	0	0	VRM	TRA MIA
16148		2	2	0	2	31	5	0	14	0	12	0	0	0	0	0	DCA	KWI
16149		1	0	1	0	11	3	2	6	0	0	0	0	0	0	0	JAU	UCA
16155		3	3	1	0	43	4	4	12	23	0	0	0	0	0	0	CLF	UCA
16155		4	2	0	4	74	5	17	12	0	12	0	0	0	0	0	HGM	PIC
16155		1	1	1	0	16	3	1	4	0	0	0	0	0	0	0	JFK	HGM
16159		4	3	3	3	75	10	19	10	14	22	0	0	0	0	0	CWM	HGM
1710		9	4	5	0	91	9	3	20	59	0	0	0	0	0	0	MSP	HGM
1712		2	2	2	2	36	6	2	8	14	6	0	0	0	0	0	MSP	HGM
1712		2	2	0	2	30	6	2	12	0	10	1	0	0	0	0	DCA	AMX
1715		1	1	0	0	9	4	1	10	0	0	0	0	0	0	0	DCA	AMX
1715		1	0	0	0	7	2	5	4	0	0	0	0	0	0	0	DCA	AMX
1715		1	1	1	0	23	4	2	8	9	0	0	0	0	0	0	PHL	AMX
1717		1	0	1	0	15	4	2	0	9	0	0	0	0	0	0	PHL	AMX
1718		7	7	7	3	119	12	13	35	45	14	0	0	0	0	0	USM	AMX
1718		2	0	2	2	43	6	2	0	20	14	0	0	0	0	0	CWM	DAY
17111		4	0	4	4	65	3	14	0	30	14	0	0	0	0	0	CMP	AGC
17111		1	1	0	0	17	4	0	9	0	0	0	0	0	0	0	CMP	AGC
17111		11	1	4	3	152	7	9	64	31	41	0	0	0	0	0	MIA	AMX
17112		1	1	0	0	11	4	1	6	0	0	0	0	0	0	0	CWM	USU
17113		1	1	0	0	17	4	2	11	0	0	0	0	0	0	0	MCO	AMX
17113		1	1	0	0	17	4	2	11	0	0	0	0	0	0	0	CWM	AMX
17115		3	3	0	0	24	7	1	20	0	0	0	0	0	0	0	GJT	GUC
17116		1	1	0	0	11	3	2	6	0	0	0	0	0	0	0	GUC	AMX
17117		4	1	4	4	70	4	3	7	30	22	0	0	0	0	0	IAD	HNL
17118		2	2	2	0	52	6	1	22	23	0	3	0	0	0	0	CWM	PIT
17120		1	1	1	0	23	4	2	7	10	0	0	0	0	0	0	CWM	AMX
17121		1	1	1	1	24	4	1	5	11	7	0	0	0	0	0	IAD	AMX
17122		5	0	5	0	67	3	14	0	46	0	0	0	0	0	0	MFO	FJA
17123		1	1	1	0	24	3	5	6	10	0	0	0	0	0	0	CWM	AMX
17124		3	1	3	0	46	4	4	3	6	13	0	0	0	0	0	IAD	MSP
17125		3	3	1	0	34	4	6	24	4	0	0	0	0	0	0	PJA	SAV

only the 1st & last page of the lengthy user report is included in this sample run.

VMS STATISTICS

MINIMUM CHANNEL

MAXIMUM CHANNEL 20

STANT DATE PD

FINISH DATE RD

1 BYTE 0 TIME 993

BYTE 262 TIME 646

CNT	TIME	COUNT		CF	HOURS (0.1 MIN)				COUNT SECT. FACT				LOC TO REQUESTED						
		LOC	SA		TH	TA	SA	FI	TH	TA	LOC	WAF	WFO	WOT	LOC	WAF	WFO	WOT	
5144	4	1	0	0	50	6	4	27	11	0	0	0	0	0	2	W2	PTS	OPS	LAX
5146	14	2	4	0	147	11	32	15	74	16	0	0	0	0	1	2	CAN	ESP	PTS
5148	2	0	0	0	14	6	1	12	0	0	0	0	0	0	0	0	LAX	LAX	
6127	2	0	0	0	1	5	1	11	4	0	0	0	0	0	0	1	AND	W2	
6128	1	0	0	0	11	3	2	0	0	0	0	0	0	0	0	0	LOC		
6116	1	1	0	0	14	3	1	5	5	0	0	0	0	0	0	0	LAX	WAF	
6120	2	2	0	2	10	0	1	10	0	15	0	0	0	0	0	0	LAX	WAF	
6123	5	5	5	5	137	20	21	31	35	21	3	0	0	0	1	1	1	1	1
6124	1	0	0	0	17	3	14	0	0	0	0	0	0	0	1	0	1	1	1
6136	5	5	0	0	106	9	41	21	3	0	0	0	0	0	0	4	LAX	LAX	LAX
6155	7	7	5	5	150	1	105	13	3	21	16	0	0	0	0	0	0	0	0
7124	7	6	1	0	50	14	2	30	4	0	0	0	0	0	0	0	0	0	0
7128	6	6	6	6	13	9	4	5	7	4	2	0	0	0	0	0	0	0	0
7110	3	2	2	3	69	5	9	6	17	29	3	0	0	0	0	0	0	0	0
7121	14	14	0	0	152	20	3	47	42	0	2	0	0	0	0	0	0	0	0
7145	5	5	0	0	152	12	2	124	10	0	2	0	0	0	0	0	0	0	0
8152	4	4	0	0	49	3	4	24	26	0	0	0	0	0	0	0	0	0	0
9116	3	3	0	3	49	14	1	21	1	13	0	0	0	0	0	0	0	0	0
9151	2	2	1	0	26	4	1	15	3	0	0	0	0	0	0	0	0	0	0
9155	2	2	1	0	36	4	2	14	16	0	0	0	0	0	0	0	0	0	0
10133	3	3	0	0	41	13	1	12	15	0	0	0	0	0	0	0	0	0	0
10135	1	1	0	0	101	9	1	4	0	0	0	0	0	0	0	0	0	0	0
10116	3	3	0	0	36	7	9	20	0	0	0	0	0	0	0	0	0	0	0
10127	2	2	2	0	42	7	1	10	31	0	0	0	0	0	0	0	0	0	0
10130	2	2	0	0	14	4	1	6	0	0	0	0	0	0	0	0	0	0	0
10134	7	7	0	0	44	9	5	47	14	0	2	0	0	0	0	0	0	0	0
10139	1	1	0	0	16	2	4	6	0	0	0	0	0	0	0	0	0	0	0
10140	1	1	0	1	27	6	4	16	0	1	0	0	0	0	0	0	0	0	0
10143	1	1	1	0	19	3	1	12	12	0	0	0	0	0	0	0	0	0	0

APPENDIX C - PROGRAM LISTING

FORTRAN IV

VOIC-03A

```

      C      DATE DEC. 12,1974 SENT TO MITRE STATO.FOR
0001      COMMON /TERM/ USTNM
0002      COMMON /OPTION/ IOPT
0003      COMMON /READ/CUREC,PPREC,PEPHI
0004      INTEGER*2 USTNM,TIME,XCHAN1,YCHAN1
0005      INTEGER*2 CUPEC(20,32),PPREC(20,32),PEPHI(20)
0006      INTEGER*2 EUP,REC(32)
0007      LOGICAL*1 IPEC(64)
0008      EQUIVALENCE(REC(1),IREC(1))
0009      DATA IREC /64*0/
0010      PRINT 7777,
0011 7777      FORMAT(1RX,' VRS STATISTICS  ')
0012      TYPE 657,
0013 657      FORMAT(' TYPE OPTION IN 12 FORMAT  ')
0014      ACCEPT 659,IUP1
0015 633      CONTINUE
0016 659      FORMAT(12)
0017 654      FORMAT(' TYPE IN MINIMUM CHANNEL NUMBER  ')
0018 654      FORMAT(' TYPE IN MAXIMUM CHANNEL NUMBER  ')
0019      TYPE 654,
0020      ACCEPT 659,XCHAN1
0021      PRINT 666,XCHAN1
0022 666      FORMAT(' MINIMUM CHANNEL  ',12)
0023      IF(XCHAN1.LE.0)XCHAN1=1
0025      TYPE 655,
0026      ACCEPT 659,YCHAN1
0027      PRINT 6667,YCHAN1
0028 6667      FORMAT(' MAXIMUM CHANNEL  ',12)
0029      IF(YCHAN1.GT.20)YCHAN1=1
0031      IF(XCHAN1.GT.YCHAN1) GO TO 633
0033 79      CONTINUE
0034      CALL READER (IREC,JJREC,FOF)
      PRINT 7766,(IPEC(10),IU=1,JJREC)
0035 776      FORMAT(1X,1406,2341)
0036      IF(FOF.GT.0)GO TO 39
0038      IUSER=IREC(1)+1
0039      IF(IUSER.LT.XCHAN1.OR.IUSER.GT.YCHAN1)GOTO 76
      PRINT 7768,IUSER
0041 7768      FORMAT(' IUSER  ',15)
0042      DO 45 J=1,32
0043      PPREC(IUSER,J)=CUREC(IUSER,J)
0044      CURFC(IUSER,J)=REC(J)
0045 45      CONTINUE
0046 52      CONTINUE
0047      CALL MINUTE(CUREC(IUSER,6),CUREC(IUSER,7),TIME,ITYPE)
0048      CALL LOCN (IUSER,JJREC,IREC)
      C      CALL KEYV (IUSER,IREC(6),IREC(7),TIME)
0049      CALL MORE(IUSER,JJREC,IREC)
0050      CALL UNAX(IUSER,IREC(6),IREC(7),TIME)
0051      CALL REPORT (IUSER,JJREC,IREC)
0052      CALL KEYI(ITYPE,IREC(6),IREC(7),IUSER)
0053      CALL STATUS (IUSER,IREC(3),IREC(7))
0054      CALL TIMER(IUSER,IREC(6),IREC(7),ITYPE)
0055      GO TO 79
0056 39      CONTINUE
0057      TYPE 789,
0058 789      FORMAT('1')
0059      CALL SUMMAR
0060      CALL SUMLOC
0061      STOP 'INIT'
0062 76      CONTINUE
0063 75      FORMAT(' CHANNEL OUT OF RANGE  ',07)
0064      GO TO 79
0065      END

```

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
REC	000006	INTEGER*2 ARRAY (32)
IREC	000006	LOGICAL*1 ARRAY (64)
TIME	000572	INTEGER*2 VARIABLE
XCHAN1	000574	INTEGER*2 VARIABLE
YCHAN1	000576	INTEGER*2 VARIABLE
EOF	000600	INTEGER*2 VARIABLE
XCHAN	000602	REAL*4 VARIABLE
READER	000600	REAL*4 PROCEDURE
JJREC	000606	INTEGER*2 VARIABLE
IUSER	000610	INTEGER*2 VARIABLE
J	000612	INTEGER*2 VARIABLE
MINUTE	000000	INTEGER*2 PROCEDURE
ITYPE	000614	INTEGER*2 VARIABLE
LOCO	000000	INTEGER*2 PROCEDURE
MORE	000000	INTEGER*2 PROCEDURE
UMAX	000000	REAL*4 PROCEDURE
REPORT	000000	REAL*4 PROCEDURE
KEYT	000000	INTEGER*2 PROCEDURE
STATUS	000000	REAL*4 PROCEDURE
TIMER	000000	REAL*4 PROCEDURE
SUMMAR	000000	REAL*4 PROCEDURE
SUMLOC	000000	REAL*4 PROCEDURE

COMMON BLOCK /TERM/ LENGTH 000002

USTPM 000000 INTEGER*2 VARIABLE

COMMON BLOCK /OPTION/ LENGTH 000002

IOPT 000000 INTEGER*2 VARIABLE

COMMON BLOCK /READ/ LENGTH 005050

CUREC	000000	INTEGER*2 ARRAY (20,32) VECTORED
PRPEC	002400	INTEGER*2 ARRAY (20,32) VECTORED
REPHI	005000	INTEGER*2 ARRAY (20)

FORTRAN IV VOIC-03A

```

0001      SUBROUTINE KEYV(IUSER,IREC6,IREC7,TIME)
0002      INTEGER*2 COUNT(20),TIME,ATIME,IMC(20)
0003      LOGICAL*1 IREC6,IREC7,KEYV(20),JREC(20)
0004      DATA JREC /20*0/
0005      DATA KEYV /20*0/
0006      DATA ATIME /0/
0007      DATA COUNT /20*0/
0008      DATA IMC /20*0/
0009      IF(IREC7.EQ.5) COUNT(3)=COUNT(3)+1
0011      IF(IREC7.EQ.6)COUNT(6)=COUNT(6)+1
0013      IF( IMC(IUSER).GT.0) GO TO 20
0015      IF(IREC7.EQ.6.AND.IREC6.NE.0)COUNT(7)=COUNT(7)+1
0017      JREC(IUSER)=IREC7
0018      IF(IREC7.NE.5) RETURN
0020      COUNT(1)= COUNT(1) +1
0021      IMC(IUSER)=1
0022      KEYV(IUSER)=0
0023      RETURN
0024 20      IF(IREC6.GT.KEYV(IUSER)) KEYV(IUSER)=IREC6
0026      IF(IREC7.EQ.5)COUNT(4)=COUNT(4)+1
0028      IF(IREC7.NE.6)JREC(IUSER)=IREC7
0030      IF(IREC7.NE.6) RETURN
0032      IF(JREC(IUSER).EQ.5)COUNT(5)=COUNT(5)+1
0034      IF(KEYV(IUSER).LE.5) COUNT(2)=COUNT(2) +1
0036      KEYV(IUSER)=0
0037      IMC(IUSER)=0
0038 78      FORMAT(1X,'      COUNT OF NON REPORT REQUESTS ',6I7,F6.4)
0039      IF((TIME-ATIME).LT.60) RETURN
0041      RATIO =(100.*COUNT(2))/COUNT(1)
0042      PRINT 78,COUNT(1),COUNT(2),COUNT(3),COUNT(4),COUNT(5),COUNT(6)
0043      PRINT 78,COUNT(7)
0044      ATIME=TIME
0045      RETURN
0046      END

```

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
COUNT	000024	INTEGER*2 ARRAY (20)
IMC	000074	INTEGER*2 ARRAY (20)
KEYV	000144	LOGICAL*1 ARRAY (20)
JREC	000170	LOGICAL*1 ARRAY (20)
IUSER	000014	INTEGER*2 PARAMETER VARIABLE
IREC6	000016	LOGICAL*1 PARAMETER VARIABLE
IREC7	000020	LOGICAL*1 PARAMETER VARIABLE
TIME	000022	INTEGER*2 PARAMETER VARIABLE
ATIME	000214	INTEGER*2 VARIABLE
RATIO	000272	REAL*4 VARIABLE

FORTRAN IV VOIC-03A

```

0001      SUBROUTINE SUMLOC
0002      COMMON /NALOC/LOC4,LOC
0003      COMMON /OPTION/ IOPT
0004      INTEGER*2 LOC4(400)
0005      LOGICAL*1 LOC(3,400),ZERO
0006      JOPT =IOPT.AND.4
0007      IF(JOPT.LE.0)RETURN
0008      PRINT 100,
0009      PRINT 101,
0010      FORMAT('1  IDENTITY CODES AND REQUEST COUNTS ')
0011  100  FORMAT('      LOC COUNT      LOC COUNT      LOC COUNT      LOC COUNT ')
0012  101  FORMAT('1' LOC COUNT      LOC COUNT ')
0013      ZERO=0
0014      IZ=0
0015  78   CONTINUE
0016      IX=IZ+1
0017      IF(IX.GE.400.OR.LOC4(IX).LT.ZERO)RETURN
0018      IP=IX+5
0019      DO 80 JR=IX,IP
0020      IF(LOC4(JR).LT.0)GO 1085
0021      IZ=IZ+1
0022      CONTINUE
0023      80
0024      85   CONTINUE
0025      76   FORMAT(6 (5X,3A1,I5))
0026      PRINT 76,((LOC(IU,II),IU=1,3),LOC4(II),II=IX,IZ)
0027      GO TO 78
0028      END
0029

```

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
ZERO	000232	LOGICAL*1 VARIABLE
JOPT	000234	INTEGER*2 VARIABLE
IZ	000236	INTEGER*2 VARIABLE
IX	000240	INTEGER*2 VARIABLE
IP	000242	INTEGER*2 VARIABLE
IR	000244	INTEGER*2 VARIABLE
IU	000246	INTEGER*2 VARIABLE
II	000250	INTEGER*2 VARIABLE

COMMON BLOCK /NALOC/ LENGTH 003720

LOC4 000000 INTEGER*2 ARRAY (400)
LOC 001440 LOGICAL*1 ARRAY (3,400)

COMMON BLOCK /OPTION/ LENGTH 000002

IOPT 000000 INTEGER*2 VARIABLE

FORTRAN IV

VOIC-03A

```

0001      SUBROUTINE MINUTE(CUR6,CUR7,TIME,ITYME)
0002          INTEGER*2 CUR6,CUR7,OCT(17),TIME
0003          DATA OCT(17) /"177777 /
0004          DATA OCT(16) /"100000 /
0005          DATA OCT(15) /"40000/
0006          DATA OCT(14) /"20000/
0007          DATA OCT(13) /"10000/
0008          DATA OCT(12) /"4000/
0009          DATA OCT(11) /"2000/
0010          DATA OCT(10) /"1000/
0011          DATA OCT(9) /"400/
0012          DATA OCT(8) /"200/
0013          DATA OCT(7) /"100/
0014          DATA OCT(6) /"40/
0015          DATA OCT(5) /"20/
0016          DATA OCT(4) /"10/
0017          DATA OCT(3) /"4/
0018          DATA OCT(2) /"2/
0019          DATA OCT(1) /"1/
0020          TIME=0
0021          IA=CUR7.AND.OCT(1)
0022          IF(IA.LE.0) GO TO 27
0024          TIME=TIME.OR.OCT(15)
0025 27      CONTINUE
0026          DO 25 I=3,16
0027          IA=CUR6.AND.OCT(I)
0028          IF(IA.EQ.0) GO TO 25
0030          TIME=TIME.OR.OCT(I-2)
0031 25      CONTINUE
0032          ITYME=(40./60.)*TIME
0033          TIME=(4./60.)*TIME
0034          RETURN
0035          END

```

FORTRAN IV

STORAGE MAP

NAME	OFFSET	ATTRIBUTES
OCT	000024	INTEGER*2 ARRAY (17)
CUR6	000016	INTEGER*2 PARAMETER VARIABLE
CUR7	000016	INTEGER*2 PARAMETER VARIABLE
TIME	000020	INTEGER*2 PARAMETER VARIABLE
ITYME	000022	INTEGER*2 PARAMETER VARIABLE
IA	000066	INTEGER*2 VARIABLE
I	000070	INTEGER*2 VARIABLE

FORTRAN IV

V01C-03A

```

0001      SUBROUTINE TIMEH (IUSER,IREF6,IREF7,TIME)
0002      COMMON /SUM/F1IME,ICO
0003      COMMON /OPTION/IOPT
0004      COMMON /TYPE1/ ITIME,JTIME,GTIME,JCO,ETIME
0005      INTEGER*7 ST(20),GTIME
0006      INTEGER*2 TIME,FTIME(48),ICO(48),ETIME,IND(20)
0007      LOGICAL*1 IHFC6,IREF7,FLG(20)
0008      DATA IND /20*0/
0009      DATA IHFX /0/
0010      DATA LTIME /600/
0011      DATA FTIME /48*0/
0012      DATA ICO /48*0/
0013      DATA ST /20*-1/
0014      DATA FLG /20*0/
0015      IH=(TIME/600) +1+IHFX
0016      ITIME=TIME/600
0017      JTIME=(TIME -600*ITIME)/10
0018      KTIME=TIME-LTIME
0019      IF (TIME.LT.600.AND.KTIME.L1.0) IHFX=24
0020      IF (TIME.LT.600.AND.KTIME.L1.0) KTIME=TIME+14400
0021      IF (KTIME.LT.600.AND.KTIME.GE.0) GO TO 900
0022      C
0023      BREAK IN CONTINUITY
0024      DO 901 IU=1,20
0025      ST(IU)=-1
0026      IND(IU)=0
0027      901 CONTINUE
0028      IF (KTIME.LT.0) IHFX=24
0029      900 CONTINUE
0030      LTIME=TIME
0031      C
0032      LINE BECOMES ACTIVE
0033      IF (IREF7.EQ.5) GO TO 23
0034      IF (IND(IUSER).GT.0) GO TO 79
0035      FLG(IUSER)=IREF7
0036      RETURN
0037      79 CONTINUE
0038      C
0039      ACTIVE LINE
0040      IF (FLG(IUSER).EQ.5.AND.IREF7.EQ.6) GO TO 900
0041      IF (FLG(IUSER).NE.6.AND.IREF7.EQ.6) GO TO 33
0042      FLG(IUSER)=IREF7
0043      RETURN
0044      C
0045      SUBROUTINE START
0046      23 ST(IUSER)=TIME
0047      FLG(IUSER)=IREF7
0048      IND(IUSER)=1
0049      RETURN
0050      C
0051      SUBROUTINE FINISH
0052      33 GTIME=TIME-ST(IUSER)
0053      IF (GTIME.LT.0.AND.TIME.L1.600) GTIME =GTIME+14400
0054      FTIME(IH)=GTIME+FTIME(IH)
0055      ICO(IH)=ICO(IH)+1
0056      ETIME=FTIME(IH)/ICO(IH)
0057      FLG(IUSER)=IREF7
0058      JCO=ICO(IH)
0059      JOPT=IOPT.AND.1
0060      IF (JOPT.LE.0) GO TO 900
0061      CALL TYPER (IUSER)
0062      800 CONTINUE
0063      76 FORMAT(1X,I2,' ',I2,4I7)
0064      ST(IUSER)=-1
0065      IND(IUSER)=0
0066      RETURN
0067      END

```

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
ST	000024	INTEGER*2 ARRAY (20)
IND	000074	INTEGER*2 ARRAY (20)
FLG	000144	LOGICAL*1 ARRAY (20)
IUSER	000014	INTEGER*2 PARAMETER VARIABLE
IREF	000016	LOGICAL*1 PARAMETER VARIABLE
IREF	000020	LOGICAL*1 PARAMETER VARIABLE
TIME	000022	INTEGER*2 PARAMETER VARIABLE
INEX	000170	INTEGER*2 VARIABLE
LTIME	000172	INTEGER*2 VARIABLE
IN	000210	INTEGER*2 VARIABLE
KTIME	000212	INTEGER*2 VARIABLE
IU	000214	INTEGER*2 VARIABLE
JOPT	000216	INTEGER*2 VARIABLE
TYPER	000000	REAL*4 PROCEDURE

COMMON BLOCK /SUM/ LENGTH 000300

FTIME	000000	INTEGER*2 ARRAY (4)
ICO	000140	INTEGER*2 ARRAY (4)

COMMON BLOCK /OPTION/ LENGTH 000002

IOPT	000000	INTEGER*2 VARIABLE
------	--------	--------------------

COMMON BLOCK /TYPE1/ LENGTH 000012

ITIME	000000	INTEGER*2 VARIABLE
JTIME	000002	INTEGER*2 VARIABLE
GTIME	000004	INTEGER*2 VARIABLE
JCO	000006	INTEGER*2 VARIABLE
ETIME	000010	INTEGER*2 VARIABLE

FORTRAA 1V

VOIC-03A

```

0001      SUBROUTINE LOCC(IUSER,JJREC,IREC)
0002      COMMON /OPTION/ IOPT
0003      COMMON /ENT5/ENT1,ENT2,ENT3
0004      COMMON /TYPE2/ COUNT
0005      INTEGER*2 COUNT(20),ENT4(20)
0006      LOGICAL*1 IREC(64),PP,MM,AA,ZZ,SLASH
0007      LOGICAL*1 ENT1(20,10),ENT2(20,10),ENT3(20,10),EL
0008      LOGICAL*1 LENT1,LENT2,LENT3
0009      DATA ENT4 /20*0/
0010      DATA PP /'P'/
0011      DATA BL /' '/
0012      DATA MM /'M'/
0013      DATA AA /'A'/
0014      DATA ZZ /'Z'/
0015      DATA SLASH /'057/'
0016      IF(IREC(7).NE.5) GO TO 199
0018      DO 94 IVV=1,10
0019      ENT1(IUSER,IVV)=BL
0020      ENT2(IUSER,IVV)=BL
0021      ENT3(IUSER,IVV)=PI
0022 99      CONTINUE
0023      ENT4(IUSER)=0
0024 199      CONTINUE
0025      IF(IREC(6).NE.1.AND.IREC(6).NE.5) RETURN
0027      IF (IREC(14).NE.PP.AND.IREC(17).NE.PP)RETURN
0029      IF(IREC(17).NE.MM.AND.IREC(18).NE.MM)RETURN
0031      COUNT(IUSER)=0
0032 78      FORMAT(' RECOFD LOCC ',27A1)
0033      C      TYPE 78,(IREC(IU),IU=15,JJREC)
0034      IKL=18
0036      IF(IREC(18).EQ.MM) IKL=19
0037      IV=1+ENT4(IUSER)
0038 89      CONTINUE
0039      IF(IV.GT.10) IV=10
0040      C      TYPE 79,COUNT(IUSER),IREC(IKL+4)
0041      DO 97 IU=1,3
0042      KL=IKL+IU
0043      IF (IREC(KL).GT.ZZ.OR.IREC(KL).LT.AA)RETURN
0044 97      CONTINUE
0046      JOPT=IOPT.AND.4
0047      IF(JOPT.LE.0) GO TO 87
0049      LENT1=IREC(IKL+1)
0050      LENT2=IREC(IKL+2)
0051      LENT3=IREC(IKL+3)
0052      CALL LOCAT(LENT1,LENT2,LENT3)
0053 87      CONTINUE
0054      JOPT=IOPT.AND.2
0055      IF (JOPT.LE.0)GO TO 86
0057      ENT1(IUSER,IV)=IREC(IKL+1)
0058      ENT2(IUSER,IV)=IREC(IKL+2)
0059      ENT3(IUSER,IV)=IREC(IKL+3)
0060      ENT4(IUSER)=IV
0061 86      CONTINUE
0062      COUNT(IUSER)=COUNT(IUSER)+1
0063      IF(IREC(IKL+4).NE.SLASH)RETURN
0065      IV=IV+1
0066      IKL=IKL+4
0067      GO TO 89
0068      RETURN
0069      END

```

FORTRAN IV STORAGE MAP

NAME	OFFSE1	ATTRIBUTES
ENT4	000022	INTEGER*2 ARRAY (20)
IPEC	000020	LOGICAL*1 PARAMETER ARRAY (64)
IUSEP	000014	INTEGER*2 PARAMETER VARIABLE
JJREC	000016	INTEGER*2 PARAMETER VARIABLE
PP	000072	LOGICAL*1 VARIABLE
MM	000074	LOGICAL*1 VARIABLE
AA	000075	LOGICAL*1 VARIABLE
ZZ	000076	LOGICAL*1 VARIABLE
SLASH	000077	LOGICAL*1 VARIABLE
RL	000073	LOGICAL*1 VARIABLE
LENT1	000170	LOGICAL*1 VARIABLE
LENT2	000171	LOGICAL*1 VARIABLE
LENT3	000172	LOGICAL*1 VARIABLE
IVV	000174	INTEGER*2 VARIABLE
IKL	000176	INTEGER*2 VARIABLE
IV	000200	INTEGER*2 VARIABLE
IU	000202	INTEGER*2 VARIABLE
KL	000204	INTEGER*2 VARIABLE
JOPT	000206	INTEGER*2 VARIABLE
LOCA1	000000	INTEGER*2 PROCEDURE

COMMON BLOCK /OPTION/ LENGTH 000002

JOPT 000000 INTEGER*2 VARIABLE

COMMON BLOCK /ENT15/ LENGTH 001130

ENT1	000000	LOGICAL*1 ARRAY (20,10) VECTORED
ENT2	000310	LOGICAL*1 ARRAY (20,10) VECTORED
ENT3	000620	LOGICAL*1 ARRAY (20,10) VECTORED

COMMON BLOCK /TYPE2/ LENGTH 000050

COUNT 000000 INTEGER*2 ARRAY (20)

FORTRAN IV VOIC-03A

```

0001        SUBROUTINE TYPEX (IUSER)
0002        COMMON /ENTS/ ENT1,ENT2,ENT3
0003        COMMON /TYPE7/TIMEX,FRACT
0004        COMMON /SPEC1/ ISPRC(20,12)
0005        COMMON /TYPE3/ FTI,SAI,FDI
0006        COMMON /TYPE2/COUNT
0007        COMMON /TYPE1/ITIME,JTIME,GTIME,JCU,FTIME
0008        INTEGER*2 FRACT(19),TIMEX(19,20), FTI(20),SAI(20),FDI(20)
0009        INTEGER*2 GTIME,JCU,ETIME,COUNT(20),LINE
0010        LOGICAL*1 ENT1(20,10),ENT2(20,10),ENT3(20,10)
0011        DATA LINE /45/
0012        IF( LINE.LT.40) GO TO 900
0013        PRINT 749,
0014        PRINT 788,
0015        LINE=2
0016        900        CONTINUE
0017        PRINT 76,ITIME,JTIME,COUNT(IUSER)
0018        1        ,SAI(IUSER),FTI(IUSER),FDI(IUSER),GTIME,TIMEX
0019        2        (1,IUSER),TIMEX(2,IUSER),
0020        2        TIMEX(14,IUSER),TIMEX(16,IUSER),TIMEX(19,IUSER),
0021        7        ISPRC(IUSER,3),ISPRC(IUSER,4),ISPRC(IUSER,6),ISPRC(IUSER,4)
0022        8        ISPRC(IUSER,7),ISPRC(IUSER,5)
0023        5        ,((ENT1(IUSER,IV),ENT2(IUSER,IV),ENT3(IUSER,IV)),IV=1,10)
0024        LINE = LINE +1
0025        76        FORMAT(1X,I2,' ',I2,2X,4I4,5X,6I5,4X,6I5,10(3A1,1X))
0026        788        FORMAT('IGMT',3X,5X,'COUNT',6X,20X,'TIMES (0.1 MIN)',
0027        110X,' COUNT SPEC. FACT')
0028        788        FORMAT(1X' TIME',2X,'LOC SA F1
0029        1        GF ',5X,' TIME OVER LOC SA FT GF ',4X,
0030        'STOP GO NPI DEL JMP REG LOC ID REQUESTED ')
0031        RETURN
0032        END

```

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
------	--------	------------

IUSER	000014	INTEGER*2 PARAMETER VARIABLE
LINE	000016	INTEGER*2 VARIABLE
IV	000442	INTEGER*2 VARIABLE

COMMON BLOCK /EN15/ LENGTH 001130

ENT1	000000	LOGICAL*1 ARRAY (20,10) VECTORED
ENT2	000310	LOGICAL*1 ARRAY (20,10) VECTORED
ENT3	000620	LOGICAL*1 ARRAY (20,10) VECTORED

COMMON BLOCK /TYPE7/ LENGTH 001430

TIMEX	000000	INTEGER*2 ARRAY (10,20) VECTORED
FRACT	001370	INTEGER*2 ARRAY (10)

COMMON BLOCK /SPEC1/ LENGTH 000740

ISPRC	000000	INTEGER*2 ARRAY (20,12) VECTORED
-------	--------	----------------------------------

COMMON BLOCK /TYPE3/ LENGTH 000170

FTI	000000	INTEGER*2 ARRAY (20)
SAI	000050	INTEGER*2 ARRAY (20)
FDI	000120	INTEGER*2 ARRAY (20)

COMMON BLOCK /TYPE2/ LENGTH 000050

COUNT	000000	INTEGER*2 ARRAY (20)
-------	--------	----------------------

COMMON BLOCK /TYPE1/ LENGTH 000012

ITIME	000000	INTEGER*2 VARIABLE
JTIME	000002	INTEGER*2 VARIABLE
GTIME	000004	INTEGER*2 VARIABLE
JCD	000006	INTEGER*2 VARIABLE
ETIME	000010	INTEGER*2 VARIABLE

FORTHAN IV

VOIC-03A

```

0001 SUBROUTINE REPORT(IUSER,JJREC,IREC)
0002 COMMON /TYPE3/FTI,SAI,FDI
0003 COMMON /REPT1/COUNTL
0004 INTEGER*2 COUNTL(20)
0005 INTEGER*2 FTI(20),SAI(20),FDI(20)
0006 LOGICAL*4 IREC(64),FF,SS,TT,AA,CU,FI,SA,FD,REC(20)
0007 DATA FTI /20*0/
0008 DATA SAI /20*0/
0009 DATA FDI /20*0/
0010 DATA REC6 /20*0/
0011 DATA FI /15/
0012 DATA SA /13/
0013 DATA FD /18/
0014 DATA FF /'F'/
0015 DATA SS /'S'/
0016 DATA TT /'T'/
0017 DATA AA /'A'/
0018 DATA DD /'D'/
0019 IF(IREC(7).EQ.5) GO TO 100
0021 IF(REC6(IUSER).EQ.IREC(6)) RETURN
0023 REC6(IUSER)=IREC(6)
0024 IF(IREC(16).NE.FF.AND.IREC(17).NE.FF)GO TO 300
0026 IF(IREC(17).NE.IT.AND.IREC(18).NE.TT) GO TO 300
0028 IF(IREC(6).EQ.FI)FTI(IUSER)=FTI(IUSER)+COUNTL(IUSER)
0030 300 CONTINUE
0031 IF(IREC(16).NE.SS.AND.IREC(17).NE.SS) GO TO 200
0033 IF(IREC(17).NE.AA.AND.IREC(18).NE.AA) GO TO 200
0035 IF(IREC(6).EQ.SA)SAI(IUSER)=SAI(IUSER)+COUNTL(IUSER)
0037 200 CONTINUE
0038 IF(IREC(16).NE.FF.AND.IREC(17).NE.FF)RETURN
0040 IF(IREC(17).NE.DD.AND.IREC(18).NE.DD)RETURN
0042 IF(IREC(6).EQ.FD)FDI(IUSER)=FDI(IUSER)+COUNTL(IUSER)
0044 RETURN
0045 100 CONTINUE
0046 FTI(IUSER)=0
0047 SAI(IUSER)=0
0048 FDI(IUSER)=0
0049 RETURN
0050 END

```

FORTHAN IV STORAGE MAP

NAME OFFSET ATTRIBUTES

IREC	000020	LOGICAL*1 PARAMETER ARRAY (64)
REC6	000022	LOGICAL*1 ARRAY (20)
IUSER	000014	INTEGER*2 PARAMETER VARIABLE
JJREC	000016	INTEGER*2 PARAMETER VARIABLE
FF	000051	LOGICAL*1 VARIABLE
SS	000052	LOGICAL*1 VARIABLE
TT	000053	LOGICAL*1 VARIABLE
AA	000054	LOGICAL*1 VARIABLE
DD	000055	LOGICAL*1 VARIABLE
FT	000046	LOGICAL*1 VARIABLE
SA	000047	LOGICAL*1 VARIABLE
FD	000050	LOGICAL*1 VARIABLE

COMMON BLOCK /TYPE3/ LENGTH 000170

FTI	000000	INTEGER*2 ARRAY (20)
SAI	000050	INTEGER*2 ARRAY (20)
FDI	000120	INTEGER*2 ARRAY (20)

COMMON BLOCK /REPT1/ LENGTH 000050

COUNTL	000000	INTEGER*2 ARRAY (20)
--------	--------	----------------------

FORTRAN IV V01C-03A

```

0001      SUBROUTINE MORE(IUSER,JJREC,IPEC)
0002      COMMON /TYPE2/COUNT
0003      COMMON /REPT1/COUNTL
0004      INTEGER*2 INC(20),COUNT(20),COUNTL(20),COUNTM(20)
0005      LOGICAL*1 IREC(64)
0006      DATA COUNTM /20*0/
0007      DATA INC /20*0/
0008      DATA COUNTL /20*0/
0009      IF(IREC(7).EQ.5) GO TO 100
0011      IF(IREC(6).NE.1.AND.IREC(6).NE.5)GO TO 700
0013      IF(INC(IUSER).LE.0)GO TO 200
0015      COUNTM(IUSER)= COUNTM(IUSER)+COUNTL(IUSER)
0016      COUNTL(IUSER)=0
0017      200      CONTINUE
0018      INC(IUSER)=0
0019      IF(COUNT(IUSER).GT.COUNTL(IUSER)) COUNTL(IUSER)=COUNT(IUSER)
0021      COUNT(IUSER)= COUNTM(IUSER) + COUNTL(IUSER)
0022      RETURN
0023      700      CONTINUE
0024      INC(IUSER)=1
0025      COUNT(IUSER)=COUNT(IUSER)+COUNTL(IUSER)
0026      RETURN
0027      100      COUNTM(IUSER)=0
0028      COUNTL(IUSER)=0
0029      COUNT(IUSER)=0
0030      INC(IUSER)=0
0031      RETURN
0032      END

```

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
INC	000022	INTEGER*2 ARRAY (20)
COUNTM	000072	INTEGER*2 ARRAY (20)
IREC	000020	LOGICAL*1 PARAMETER ARRAY (64)
IUSER	000014	INTEGER*2 PARAMETER VARIABLE
JJREC	000016	INTEGER*2 PARAMETER VARIABLE

COMMON BLOCK /TYPE2/ LENGTH 000050

COUNT 000000 INTEGER*2 ARRAY (20)

COMMON BLOCK /REPT1/ LENGTH 000050

COUNTL 000000 INTEGER*2 ARRAY (20)

FORTRAN IV VOIC-03A

```

0001      SUMMOUTIME KEYI(TIME,IREC6,IREC7,IUSER)
0002      COMMON /TYPE7/TYPEK,FRACT
0003      INTEGER*2 TIME,KEYI(19),KEYU(20),TIMEK(19,20),FRACT(19),S1(20)
1       ,SE(20),ITIME(20),IMC(20)
0004      LOGICAL*1 IREC6,IREC7
0005      DATA S1 /20*0/
0006      DATA IMC /20*0/
0007      DATA IXYZ /977/
0008      DATA SE /20*0/
0009      DATAKEYU /20*18/
0010      DATA FRACT /19*0/
0011      DATA ITIME /20*0/
0012      IF(IXYZ.NE.977) GO TO 977
0014      DO 200 IU=1,19
0015      DO 201 JU=1,20
0016      TIMEK(IU,JU)=0
0017 201   CONTINUE
0018      KEYI(IU)=IU
0019 200   CONTINUE
0020      KEYI(6)=2
0021      KEYI(18)=19
0022      KEYI(17)= 19
0023 977   CONTINUE
0024      IXYZ=0
0025      IF(IREC7.NE.5) GO TO 300
0027      IMC(IUSER)=2
0028      SE(IUSER)=TIME
0029      ST(IUSER)=TIME
0030      ITIME(IUSER)=0
0031      DO 301 IU=1,19
0032      FRACT(IU)=FRACT(IU)+TIMEK(IU,IUSER)
0033      TIMEK(IU,IUSER)=0
0034 301   CONTINUE
0035 300   IF(IMC(IUSER).LE.0) RETURN
0037      IF(IREC7.EQ.6) IMC(IUSER)=0
0039      SE(IUSER)=TIME
0040      KEY=IREC6+1
0041      KEYJ=KEYI(KEY)
0042      IF(KEYJ.FQ.3.DM.KEYJ.FQ.5)KEYJ=KEYU(IUSER)
0044      ITIME(IUSER)=SE(IUSER)-S1(IUSER)
0045      IF(ITIME(IUSER).LT.0)ITIME(IUSER)=0
0047      IF(KEYJ.NE.KEYU(IUSER)) GO TO 100
0049      IF(IREC7.EQ.6) GO TO 100
0051      RETURN
0052 100   KEYL=KEYU(IUSER)
0053      TIMEK(KEYL,IUSER)=TIMEK(KEYL,IUSER)+ITIME(IUSER)
0054      ITIME(IUSER)=0
0055      ST(IUSER)=SE(IUSER)
0056      KEYU(IUSER)=KEYJ
0057      RETURN
0058      END

```


FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
KEYI	000024	INTEGER*2 ARRAY (19)
KEYU	000072	INTEGER*2 ARRAY (20)
ST	000142	INTEGER*2 ARRAY (20)
SE	000212	INTEGER*2 ARRAY (20)
ITIME	000262	INTEGER*2 ARRAY (20)
IMC	000332	INTEGER*2 ARRAY (20)
TIME	000014	INTEGER*2 PARAMETER VARIABLE
IREC6	000016	LOGICAL*1 PARAMETER VARIABLE
IREC7	000020	LOGICAL*1 PARAMETER VARIABLE
IUSER	000022	INTEGER*2 PARAMETER VARIABLE
IXYZ	000402	INTEGER*2 VARIABLE
IU	000454	INTEGER*2 VARIABLE
JU	000456	INTEGER*2 VARIABLE
KEY	000460	INTEGER*2 VARIABLE
KEYJ	000462	INTEGER*2 VARIABLE
KEYL	000464	INTEGER*2 VARIABLE

COMMON BLOCK /TYPE7/ LENGTH 001436

TIMEA	000000	INTEGER*2 ARRAY (19,20) VECTORED
FRACT	001370	INTEGER*2 ARRAY (19)

FORTRAN IV VOIC-03A

```

0001      SUMWOUTINE STATUS(IUSER,IREC3,IREC7)
0002      COMMON /SPEC1/ISPRC(20,12)
0003      LOGICAL*1 IREC3,MIN,GO,IREC7
0004      DATA GO /99/
0005      DATA MIN /40/
0006      IF(GO.NE.99) GO TO 120
0008      DO 87 IU=1,20
0009      DO 88 JU=1,12
0010      ISPRC(IU,JU)=0
0011      88      CONTINUE
0012      87      CONTINUE
0013      GO=789
0014      120      CONTINUE
0015      IVAL=IREC3-MIN + 1
0016      IF(IVAL.LT.1,OP,IVAL,CT,12) GO TO 900
0018      ISPRC(IUSER,IVAL)=ISPRC(IUSER,IVAL)+1
0019      900      CONTINUE
0020      IF(IREC7.EQ.5) GO TO 100
0022      RETURN
0023      100      CONTINUE
0024      DO 23 IU=1,12
0025      ISPRC(IUSER,IU)=0
0026      23      CONTINUE
0027      RETURN
0028      END

```

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
IUSER	000014	INTEGER*2 PARAMETER VARIABLE
IREC3	000016	LOGICAL*1 PARAMETER VARIABLE
IREC7	000020	LOGICAL*1 PARAMETER VARIABLE
MIN	000023	LOGICAL*1 VARIABLE
GO	000022	LOGICAL*1 VARIABLE
IU	000054	INTEGER*2 VARIABLE
JU	000056	INTEGER*2 VARIABLE
IVAL	000060	INTEGER*2 VARIABLE

COMMON BLOCK /SPEC1/ LENGTH 000740

ISPRC 000000 INTEGER*2 ARRAY (20,12) VECTORED

FORTRAN IV VOIC-03A

```

0001        SUBROUTINE LOCAT (ENT1,ENT2,ENT3)
0002        COMMON /NALOC/ LOC4,LOC
0003        INTEGER*2 LOC4(400)
0004        LOGICAL*1 LOC(3,400),ENT1,ENT2,ENT3
0005        DATA NLOC /0/
0006        IF(NLOC.LE.0) GO TO 10
0008        DO 1 I=1,NLOC
0009        ML=I
0010        IF(LOC(I,I).EQ.ENT1) GO TO 100
0012        1        CONTINUE
0013        10        NLOC=NLOC+1
0014        LOC4(NLOC)=1
0015        LOC4(NLOC+1)=-1
0016        LOC(1,NLOC)=ENT1
0017        LOC(2,NLOC)=ENT2
0018        LOC(3,NLOC)=ENT3
0019        C        PRINT 79,ENT1,ENT2,ENT3
0020        79        FORMAT(' LOCATION ',3A1)
0021        RETURN
0022        100       IF(LOC(2,NL).NE.ENT2.OR.LOC(3,NL).NE.ENT3) GO TO 1
0023        LOC4(NL)=LOC4(NL) + 1
0024        RETURN
***** E

```

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
ENT1	000014	LOGICAL*1 PARAMETER VARIABLE
ENT2	000016	LOGICAL*1 PARAMETER VARIABLE
ENT3	000020	LOGICAL*1 PARAMETER VARIABLE
NLOC	000022	INTEGER*2 VARIABLE
I	000046	INTEGER*2 VARIABLE
NL	000050	INTEGER*2 VARIABLE

COMMON BLOCK /NALOC/ LENGTH 003720

LOC4	000000	INTEGER*2 ARRAY (400)
LOC	001440	LOGICAL*1 ARRAY (3,400)

FORTRAN IV

V01C-03A

```

      C      DATE DEC. 12, 1979 SENT TO MITRE STATC.FOR
0001      SUBROUTINE ERROR (LREC,KLI,IERR)
0002      LOGICAL*1 LRFC(64)
0003      DATA MINUE /-14/
0004      IF(LREC(1).EQ.MINUE) GO TO NO
0005      KLI=KLI+2
0006      IERR= IERR+1
0007      TYPE 76,LREC(1),LREC(2),LRFC(3),LRFC(4)
0008      76      FORMAT(' ERROR ',4I5)
0009      RETURN
0010      IF(IERR.GT.0)TYPE 87,IERR
0011      87      FORMAT(' THERE WERE ',I5,' ERRORS AT THIS POINT ')
0012      IERR=0
0013      RETURN
0014      END
0015
0016

```

FORTRAN IV

STORAGE MAP

NAME	OFFSET	ATTRIBUTES
LREC	000014	LOGICAL*1 PARAMETER ARRAY (64)
KLI	000016	INTEGER*2 PARAMETER VARIABLE
IERR	000020	INTEGER*2 PARAMETER VARIABLE
MINUE	000022	INTEGER*2 VARIABLE

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
------	--------	------------

SICD	000540	INTEGER*2 VARIABLE
ATIME	000542	REAL*4 VARIABLE
MKEYM	000546	INTEGER*2 VARIABLE
IU	000550	INTEGER*2 VARIABLE
ETIME	000552	REAL*4 VARIABLE
JU	000556	INTEGER*2 VARIABLE
BTIME	000560	REAL*4 VARIABLE
MFRACT	000564	INTEGER*2 VARIABLE
IJ	000566	INTEGER*2 VARIABLE

COMMON BLOCK /OPTION/ LENGTH 000002

IOPT	000000	INTEGER*2 VARIABLE
------	--------	--------------------

COMMON BLOCK /TYPE7/ LENGTH 001436

TIMEX	000000	INTEGER*2 ARRAY (19,20) VECTORED
FRACT	001370	INTEGER*2 ARRAY (19)

COMMON BLOCK /SUM7/ LENGTH 000140

KEYM	000000	INTEGER*2 ARRAY (46)
------	--------	----------------------

COMMON BLOCK /SUM/ LENGTH 000300

ETIME	000000	INTEGER*2 ARRAY (48)
ICO	000140	INTEGER*2 ARRAY (48)

FORTRAN IV VOIC-03A

```

0001      SUBROUTINE UMAY(IUSER,IRFC6,IREC7,ITIME)
0002      COMMON /SUB7/ KEYM
0003      INTEGER*2 COUNT(48),KEYM(48),TIME,ATIME,IMC(20)
0004      LOGICAL*1 IREC6,IREC7
0005      DATA KEYM /48*0/
0006      DATA IJ /1/
0007      DATA ATIME /0/
0008      DATA COUNT /48*0/
0009      DATA ITEX /0/
0010      DATA IMC /20*0/
0011      IT=(TIME/60) +1
0012      IF(IT.LT.IJ)ITEX=24
0013      IT=IT+ITEX
0014      IF(IT.GT.IJ)COUNT(IJ)=COUNT(IJ)
0015      IJ=IT
0016      IF( IMC(IUSER).GT.0) GO TO 20
0017      IF(IREC7.NE.5) RETURN
0018      COUNT(IT)=COUNT(IT) +1
0019      IMC(IUSER)=1
0020      RETURN
0021 20      IF(COUNT(IJ).GT.KEYM(IJ)) KEYM(IJ)=COUNT(IT)
0022      IF(IREC7.NE.6) RETURN
0023      COUNT(IT)=COUNT(IT)-1
0024      IMC(IUSER)=0
0025 7H      FORMAT(IX,207)
0026      RETURN
0027      END

```

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
COUNT	000074	INTEGER*2 ARRAY (48)
IMC	000164	INTEGER*2 ARRAY (20)
IUSER	000014	INTEGER*2 PARAMETER VARIABLE
IRFC6	000016	LOGICAL*1 PARAMETER VARIABLE
IREC7	000020	LOGICAL*1 PARAMETER VARIABLE
TIME	000022	INTEGER*2 PARAMETER VARIABLE
ATIME	000234	INTEGER*2 VARIABLE
IJ	000234	INTEGER*2 VARIABLE
ITEX	000240	INTEGER*2 VARIABLE
IT	000250	INTEGER*2 VARIABLE

COMMON BLOCK /SUB7/ LENGTH 000140

KEYM 000000 INTEGER*2 ARRAY (48)

```

0001      SUBROUTINE READER(LREC,JREC,EOF)
CVPS  STATISTICS PACKAGE
0002      INTEGER*2 WCNT,RLK,CHAN,COUNT,PEC(32),EOF
0003      LOGICAL*1 BUFF(512),LREC(64),CH
0004      LOGICAL*1 PH,MM,SS,AA,FF,TT,DD,IYES,NO
0005      DATA BLK /0/
0006      DATA NU /'N'/
0007      DATA PP /'P'/
0008      DATA MM /'M'/
0009      DATA SS /'S'/
0010      DATA AA /'A'/
0011      DATA FF /'F'/
0012      DATA TT /'T'/
0013      DATA DD /'D'/
0014      DATA IERR /0/
0015      DATA LKLI /0/
0016      DATA KLI /512/
0017      DATA CHAN /-1/
0018      DATA CR /'15/
0019      DATA BLK /0/
0020      DATA WCNT /756/
0021      DATA KLI /512/
0022      DATA COUNT /0/
0023      EOF=0
CC      TYPE 980,JREC,EOF
0024  980  FORMAT(1X,517)
C      INITIALIZE
0025      IF(LALI.LT.0) GO TO 20
0027      IF(MLK.GI.0) GO TO 20
0029      CALL OPENER (CHAN)
C      TYPE 980,KLI,BLK,JREAD,LKLI,MLK,CHAN
0030      CALL INIT (KLI,BLK,JREAD,LKLI,MLK,CHAN)
C      TYPE 980 ,KLI,BLK,JREAD,LALI,MLK,CHAN
C      GET NEXT RECORD
0031  20  CONTINUE
0032      IF(BLK.GT.4MLK) GO TO 30
0034      IF(ALI.GE.LKLI.AND.BLK.GE.MLK) GO TO 30
0036      JREC=4
C      TYPE 980,KLI,BLK,JREAD,LALI,MLK
0037      CALL READS (LREC,JREC,KLI,JREAD,BLK,CHAN)
0038      IF(JHEAD.LT.0) GO TO 30
0040      CALL ERROR(LREC,KLI,IERR)
0041      IF(IERR.GT.30)GO TO 33
0043      IF(IERR.GT.25.AND.LALI.LI.0) GO TO 33
0045      IF(IERR.GT.0) GO TO 20
0047      JREC=14+LREC(3)
0048      CALL READS (LREC,JREC,KLI,JREAD,BLK,CHAN)
0049      JREC=14+JREC
C      PRINT 200,LREC(1),(LREC(10),J0=1,JREC(2)
0050      IF(LREC(16).EQ.PP.AND.LREC(17).EQ.MM)LREC(6)=5
0052      IF(LREC(16).EQ.SS.AND.LREC(17).EQ.AA) LREC(6)=13
0054      IF(LREC(16).EQ.FF.AND.LREC(17).EQ.TT) LREC(6)=15
0056      IF(LREC(16).EQ.FF.AND.LREC(17).EQ.DD) LREC(6)=16
0058      IF(LREC(1).GT.20)LREC(1)=LREC(1)/10
0060  200  FORMAT(' LREC ',4005)
0061      IF(JHEAD.LT.0) GO TO 30
0063      RETURN
0064  30  CALL CLOSFR(JREAD,CHAN,EOF)
0065      RETURN
0066  33  CONTINUE
0067      TYPE 34, IERR
0068  34  FORMAT(' THERE ARE ',15,' CONSECUTIVE ERRORS ')
0069      TYPE 36,
0070  36  FORMAT(' DO YOU WISH TO TERMINATE. TYPE YES OR NO ')
0071      ACCEPT 37,IYES
0072  37  FORMAT(A1)
0073      IF(IYES.EQ.NO) GO TO 20
0075      GO TO 30
0076      END

```


FORTHAN IV		STORAGE MAP
NAME	OFFSET	ATTRIBUTES
REC	000022	INTEGER*2 ARRAY (32)
BUFF	000122	LOGICAL*1 ARRAY (512)
LREC	000014	LOGICAL*1 PARAMETER ARRAY (64)
JREC	000016	INTEGER*2 PARAMETER VARIABLE
EOF	000020	INTEGER*2 PARAMETER VARIABLE
#CNT	001146	INTEGER*2 VARIABLE
BLK	001172	INTEGER*2 VARIABLE
CHAN	001142	INTEGER*2 VARIABLE
COUNT	001150	INTEGER*2 VARIABLE
CR	001144	LOGICAL*1 VARIABLE
PP	001125	LOGICAL*1 VARIABLE
MM	001126	LOGICAL*1 VARIABLE
SS	001127	LOGICAL*1 VARIABLE
AA	001130	LOGICAL*1 VARIABLE
FF	001131	LOGICAL*1 VARIABLE
TT	001132	LOGICAL*1 VARIABLE
DD	001133	LOGICAL*1 VARIABLE
IYES	001326	LOGICAL*1 VARIABLE
NO	001124	LOGICAL*1 VARIABLE
IERR	001134	INTEGER*2 VARIABLE
LKLI	001136	INTEGER*2 VARIABLE
KLI	001140	INTEGER*2 VARIABLE
OPENP	000000	REAL*4 PROCEDURE
INIT	000000	INTEGER*2 PROCEDURE
JREAD	001330	INTEGER*2 VARIABLE
NBLK	001332	INTEGER*2 VARIABLE
READS	000000	REAL*4 PROCEDURE
ERROR	000000	REAL*4 PROCEDURE
JRECZ	001334	INTEGER*2 VARIABLE
CLOSEP	000000	REAL*4 PROCEDURE

FORTMAN IV VOIC-03A

```

0001      SUBROUTINE CLOSER(JREAD,CHAN,EOF)
0002      INTEGER*2 CHAN,EOF
0003      IF (JREAD.LE.-2)STOP ' READ ERROR '
0004      EOF=1
0005      CALL CLOSEC(CHAN)
0006      I=IFREC(CHAN)
0007      RETURN
0008      END
0009

```

FORTMAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
JREAD	000014	INTEGER*2 PARAMETER VARIABLE
CHAN	000016	INTEGER*2 PARAMETER VARIABLE
EOF	000020	INTEGER*2 PARAMETER VARIABLE
CLOSEC	000000	REAL*4 PROCEDURE
I	000036	INTEGER*2 VARIABLE
IFREC	000000	INTEGER*2 PROCEDURE

FORTRAN IV VOIC-03A

```

0001      SUBROUTINE READS(LPEC,JREC,KLI,JREAD,BLK,CHAN)
0002          INTEGER*2 NCNT,FLK,CHAN,COUNT,REC(32)
0003      LOGICAL*1 BUFF(512),LWFC(64),CR
0004          DATA ITIME /0/
0005          DATA NPEC /0/
0006          DATA CR /'15'/
0007          DATA NCNT /256/
0008              DATA COUNT /0/
0009      NPEC=1
0010      C      GET NEXT RECORD
0011      20      IREC=512-KLI
0012      KREC=NPEC
0013          IF (IREC.GT.JREC) KREC=JREC
0014      100      FORMAT(1X,715)
0015      30      CONTINUE
0016          IF (KREC.LE.0) GO TO 54
0017      57      FORMAT('    DU LOOP')
0018      DO 35 I=NREC,KREC
0019          KLI=KLI+1
0020          IF (I.GT.64) GO TO 35
0021      LPEC(I)=BUFF(KLI)
0022      101      FORMAT(1X,315)
0023      35      CONTINUE
0024          IF (JREC.GT.64) JREC=64
0025      54      CONTINUE
0026      NPEC=NPEC+1
0027      IF (KREC.GE.JREC) RETURN
0028      JBLK=FLK
0029      IF (JBLK.GT.1000) JBLK=JBLK-1000
0030      50      JRFAD=IREAD+(NCNT,BUFF,JBLK,CHAN)
0031      IF (JREAD.LT.0) RETURN
0032      KLI=0
0033      BLK=BLK+1
0034      GO TO 70
0035      END

```

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
REC	000030	INTEGER*2 ARRAY (32)
BUFF	000130	LOGICAL*1 ARRAY (512)
LREC	000014	LOGICAL*1 PARAMETER ARRAY (64)
JREC	000016	INTEGER*2 PARAMETER VARIABLE
KLI	000020	INTEGER*2 PARAMETER VARIABLE
JREAD	000022	INTEGER*2 PARAMETER VARIABLE
BLK	000024	INTEGER*2 PARAMETER VARIABLE
CHAN	000026	INTEGER*2 PARAMETER VARIABLE
NCNT	001136	INTEGER*2 VARIABLE
COUNT	001140	INTEGER*2 VARIABLE
CR	001134	LOGICAL*1 VARIABLE
ITIME	001130	INTEGER*2 VARIABLE
NPEC	001132	INTEGER*2 VARIABLE
NREC	001170	INTEGER*2 VARIABLE
IREC	001172	INTEGER*2 VARIABLE
KREC	001174	INTEGER*2 VARIABLE
I	001176	INTEGER*2 VARIABLE
JBLK	001200	INTEGER*2 VARIABLE
IREAD	000000	INTEGER*2 PROCEDURE

FORTHAN IV

VOIC-03A

```

0001      SUBROUTINE INIT (KLI,MLA,JPEAD,LALI,MHLK,CHAN)
0002      COMMON /OPTION/ IOPT
0003      INTEGER*2 REC(32),MLK,BITS(3),TEN,RECS,YEAR,CHAN
0004      DATA TEN /20/
0005      DATA BITS /*37,*1740,*76000/
0006      TYPE 980,TEN,KLI,JPEAD,MLK,CHAN
0007 980   FORMAT(' INIT ',717)
0008      CALL READS(MFC,TEN,KLI,JPEAD,MLK,CHAN)
0009      IF(JPEAD.LT.0) RETURN
0010      CALL MINUTE(MFC(2),REC(3),ITIME,ITYME)
0011      CALL MINUTE(REC(7),REC(8),JTIME,JTYME)
0012      YEAR=(REC(1).AND.BITS(1)) +72
0013      MO=MFC(1).AND.BITS(3)
0014      MO=MO/1024
0015      IDAY=REC(1).AND.BITS(2)
0016      IDAY=IDAY/32
0017      MLA=REC(4)
0018      RECS=REC(5)
0019      IYEAR=(REC(6).AND.BITS(1)) + 72
0020      IMO=MFC(6).AND.BITS(3)
0021      IMO=IMO/1024
0022      KDAY=REC(6).AND.BITS(2)
0023      KDAY=KDAY/32
0024      LKLI=REC(10)
0025      MHLK=REC(9)
0026      IF(REC(1).GE.0) GO TO 513
0027      TYPE 514,
0028      PRINT 514,
0029 514   FORMAT(' DATA APPEARS TO START IN BLOCK ZERO ')
0030      MLA=1000
0031      RECS=0
0032      LKLI=-4
0033      MLA=0
0034 513   CONTINUE
0035      PRINT 72,YEAR,MO,IDAY,MLA,RECS,ITYME
0036      PRINT 75,IYEAR,IMO,KDAY,MHLK,LALI,JTIME
0037 72   FORMAT(' START DATE ',313,' BLK ',14,' BYTE',
0038 1 14,' TIME ',14)
0039 75   FORMAT(' FINISH DATE ',313,' BLK ',14,' BYTE ',
0040 2 14,' TIME ',14)
0041      IOPT=IOPT.AND.8
0042      IF(JOPI.LE.0) GO TO 717
0043      TYPE 780,
0044      PRINT 780,
0045 780   FORMAT(' ENTER BLOCK, BYTE')
0046      ACCEPT 765, MLA,RECS
0047 765   FORMAT(214)
0048      IF(MLK.GT.MLA) MLA=MLA+1000
0049 717   CONTINUE
0050      KLI=512
0051      CALL READS(REC,TEN,KLI,JPEAD,MLK,CHAN)
0052      KLI=512+RECS
0053      IF(KLI.GE.512) MLK = MLK+1
0054      MHLK=MHLK+1
0055      RETURN
0056      END

```

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
REC	000030	INTEGER*2 ARRAY (32)
BITS	000130	INTEGER*2 ARRAY (3)
KLI	000014	INTEGER*2 PARAMETER VARIABLE
BLK	000016	INTEGER*2 PARAMETER VARIABLE
JREAD	000020	INTEGER*2 PARAMETER VARIABLE
LKLI	000022	INTEGER*2 PARAMETER VARIABLE
NBLK	000024	INTEGER*2 PARAMETER VARIABLE
CHAN	000026	INTEGER*2 PARAMETER VARIABLE
TEN	000136	INTEGER*2 VARIABLE
RECS	000444	INTEGER*2 VARIABLE
YEAR	000446	INTEGER*2 VARIABLE
READS	000000	REAL*4 PROCEDURE
MINUTE	000000	INTEGER*2 PROCEDURE
ITIME	000450	INTEGER*2 VARIABLE
ITYME	000452	INTEGER*2 VARIABLE
JTIME	000454	INTEGER*2 VARIABLE
JTYME	000456	INTEGER*2 VARIABLE
MO	000460	INTEGER*2 VARIABLE
IDAY	000462	INTEGER*2 VARIABLE
IYEAR	000464	INTEGER*2 VARIABLE
IMO	000466	INTEGER*2 VARIABLE
KDAY	000470	INTEGER*2 VARIABLE
JOPT	000472	INTEGER*2 VARIABLE

COMMON BLOCK /OPTION/ LENGTH 000002

IOPT 000000 INTEGER*2 VARIABLE

FORTRAN IV VOIC-03A

```

0001 SUBROUTINE OPFNR(CHA)
0002 REAL*4 DBLK(2)
0003 INTEGER*2 CHAN,COUNT
0004 DATA DBLK /6RDOUTRA,APCH DAT/
0005 CHAN = IGEJC()
0006 IF(CHAN.LT.0) STOP 'CAN NOT ALLOCATE CHANNEL '
0008 IF= IFETCH (DBLK)
0009 TYPE 56,IF
0010 56 FORMAT(1X,I4)
0011 IF (IF.GT.0) STOP 'FETCH ERROR'
0013 IL = LOOKUP (CHAN,DBLK,COUNT)
0014 IF (IL.LT.0) STOP 'LOOKUP ERROR '
0016 RETURN
0017 END

```

FORTRAN IV STOPAGE MAP

NAME	OFFSET	ATTIBUTES
DBLK	000016	REAL*4 ARRAY (2)
CHAN	000014	INTEGER*2 PARAMETER VARIABLE
COUNT	000020	INTEGER*2 VARIABLE
IGEJC	000000	INTEGER*2 PROCEDURE
IF	000022	INTEGER*2 VARIABLE
IFETCH	000000	INTEGER*2 PROCEDURE
IL	000024	INTEGER*2 VARIABLE
LOOKUP	000000	INTEGER*2 PROCEDURE